aws

Guida per l'utente

Amazon EKS

Amazon EKS: Guida per l'utente

Table of Contents

Che cosa è Amazon EKS?

Amazon Elastic Kubernetes Service (Amazon EKS) è un servizio gestito che elimina la necessità di installare, gestire e mantenere un proprio piano di controllo Kubernetes su Amazon Web Services (AWS). Kubernetes è un sistema open source che automatizza la gestione, il dimensionamento e l'implementazione di applicazioni containerizzate.

Funzionalità di Amazon EKS

Di seguito sono riportate le funzionalità principali di Amazon EKS:

Rete e autenticazione sicure

Amazon EKS integra i tuoi Kubernetes carichi di lavoro con servizi AWS di rete e sicurezza. Si integra inoltre con AWS Identity and Access Management (IAM) per fornire l'autenticazione per i cluster. Kubernetes

Facile dimensionamento del cluster

Amazon EKS consente di dimensionare facilmente i cluster Kubernetes, aumentandoli e riducendoli in base alla domanda dei carichi di lavoro. Amazon EKS supporta il dimensionamento automatico orizzontale dei Pod in base alla CPU o a parametri personalizzati, nonché il dimensionamento automatico dei cluster in base alla domanda dell'intero carico di lavoro.

Esperienza Kubernetes gestita

È possibile apportare modifiche al cluster Kubernetes utilizzando `eksctl`, AWS Management Console, AWS Command Line Interface (AWS CLI), l'API, `kubectl` e Terraform.

Elevata disponibilità

Amazon EKS fornisce un'elevata disponibilità per il piano di controllo su più zone di disponibilità.

Integrazione con i servizi AWS

Amazon EKS si integra con altri servizi AWS, fornendo una piattaforma completa per l'implementazione e la gestione delle applicazioni containerizzate. Inoltre, permette di risolvere più facilmente i problemi dei carichi di lavoro Kubernetes attraverso vari strumenti per l'osservabilità.

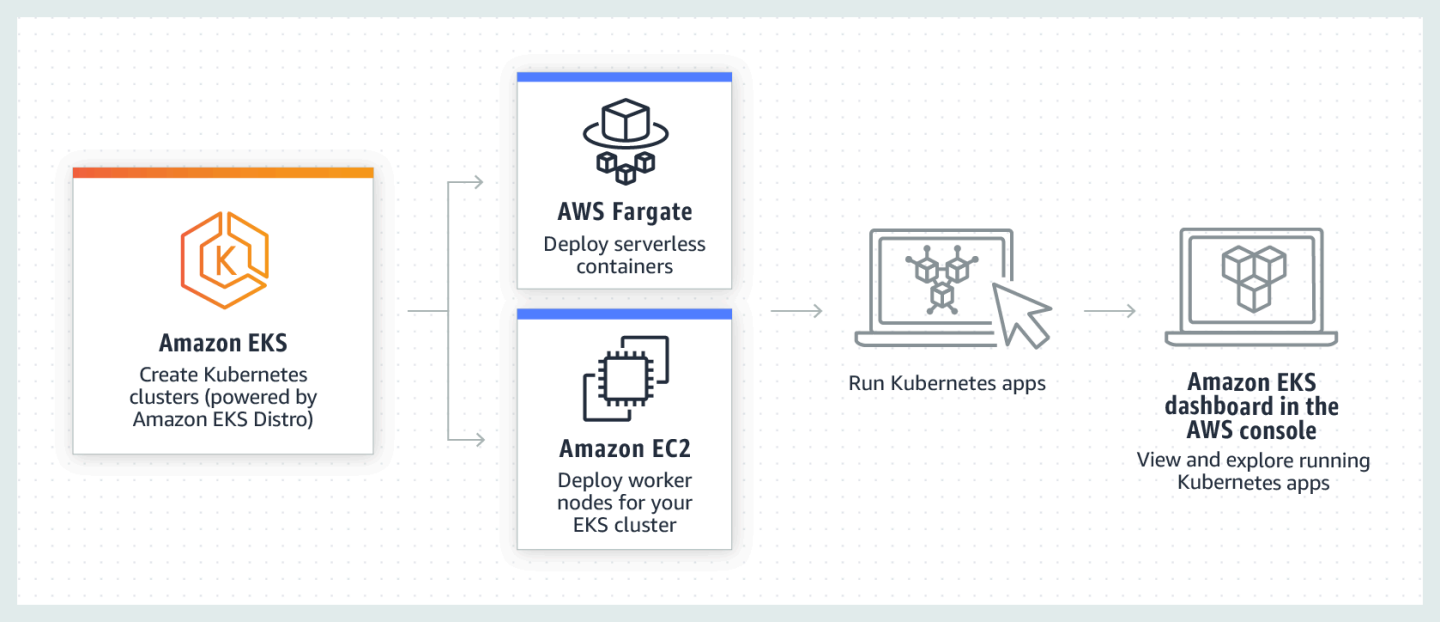
Per ulteriori dettagli sulle altre funzionalità di Amazon ELS, consulta la pagina Caratteristiche di Amazon EKS.

Nozioni di base su Amazon EKS

Per creare il primo cluster e le risorse associate, vedere Guida introduttiva ad Amazon EKS. In generale, per iniziare a utilizzare Amazon EKS è necessario compiere i seguenti passaggi.

1. Crea un cluster: inizia creando il tuo cluster utilizzando `eksctl` AWS Management Console, AWS CLI, o uno degli AWS SDK.
2. Scegli il tuo approccio alle risorse di elaborazione: decidi tra AWS Fargate gruppi di nodi gestiti e nodi autogestiti. Karpenter
3. Configurazione: configura i controller, i driver e i servizi necessari.
4. Implementazione dei carichi di lavoro: personalizza i carichi di lavoro Kubernetes per utilizzare al meglio le risorse e le funzionalità del tipo di nodo prescelto.
5. Gestione: supervisiona i carichi di lavoro integrando servizi AWS per semplificare le operazioni e migliorare le prestazioni dei carichi di lavoro. Puoi visualizzare le informazioni sui tuoi carichi di lavoro utilizzando. AWS Management Console

Il diagramma seguente mostra un flusso di base per l'esecuzione di Amazon EKS nel cloud. Per ulteriori informazioni sulle altre opzioni di implementazione di Kubernetes, consulta Opzioni di implementazione.



Prezzi di Amazon EKS

Un cluster Amazon EKS è costituito da un piano di controllo (control-plane) e dalla capacità di calcolo Amazon Elastic Compute Cloud (Amazon EC2) o Fargate su cui vengono eseguiti i Pods. Per ulteriori informazioni sui prezzi del piano di controllo, vedi Prezzi di Amazon EKS. Sia Amazon EC2 che Fargate forniscono:

Istanze on demand

Viene addebitato il costo di utilizzo delle istanze calcolato al secondo, senza impegni a lungo termine o pagamenti anticipati. Per ulteriori informazioni, consulta le pagine Prezzi di Amazon EC2 on demand e Prezzi di AWS Fargate.

, Savings Plans

È possibile ridurre i costi sottoscrivendo l'impegno a svolgere una quantità consistente di attività, in USD per ora, per un periodo di uno o tre anni. Per ulteriori informazioni, consulta la pagina Prezzi con Savings Plans.

Casi d'uso comuni in Amazon EKS

Amazon EKS offre servizi Kubernetes gestiti e affidabili su AWS, progettati per ottimizzare le applicazioni containerizzate. Di seguito sono riportati alcuni dei casi d'uso più comuni di Amazon EKS per aiutarti a sfruttarne i punti di forza per le tue esigenze specifiche.

Implementazione di applicazioni ad alta disponibilità

Utilizzando Elastic Load Balancing, puoi assicurarti che le applicazioni siano altamente disponibili in più zone di disponibilità.

Creazione di architetture di microservizi

Utilizzando la funzionalità di rilevamento servizi Kubernetes con AWS Cloud Map o Amazon VPC Lattice puoi creare sistemi resilienti.

Automazione del processo di rilascio del software

Gestisci le pipeline di integrazione e implementazione continua (CI/CD) che semplificano il processo di creazione, test e implementazione automatizzati delle applicazioni.

Esecuzione di applicazioni serverless

Utilizza AWS Fargate con Amazon EKS per eseguire applicazioni serverless. In questo modo, potrai concentrarti esclusivamente sullo sviluppo delle applicazioni mentre Amazon EKS e Fargate gestiscono l'infrastruttura sottostante.

Esecuzione di carichi di lavoro di machine learning

Amazon EKS è compatibile con i framework di machine learning più diffusi, come TensorFlow, MXNet e PyTorch. Con il supporto GPU, è possibile gestire efficacemente anche attività di machine learning complesse.

Implementazione coerente on-premise e nel cloud

Utilizza Amazon EKS Anywhere per gestire cluster Kubernetes sulla tua infrastruttura utilizzando strumenti coerenti con Amazon EKS nel cloud.

Esecuzione di carichi di lavoro di elaborazione in batch e big data a costi contenuti

Utilizza le istanze spot per eseguire carichi di lavoro di elaborazione in batch e big data, ad esempio Apache Hadoop e Spark, a una frazione del costo. Ciò ti consente di sfruttare la capacità inutilizzata di Amazon EC2 a prezzi scontati.

Protezione dell'applicazione e garanzia della conformità

Implementa pratiche di sicurezza solide e mantieni la conformità con Amazon EKS, che si integra con i servizi di sicurezza AWS come AWS Identity and Access Management (IAM), Amazon Virtual Private Cloud (Amazon VPC) e AWS Key Management Service (AWS KMS). Ciò garantisce la privacy e la protezione dei dati secondo gli standard del settore.

Architettura di Amazon EKS

Amazon EKS segue l'architettura generale dei cluster di Kubernetes. Per ulteriori informazioni, consulta la sezione Kubernetes Components nella documentazione di Kubernetes. Le seguenti sezioni riassumono alcuni dettagli aggiuntivi sull'architettura di Amazon EKS.

Piano di controllo (control-plane)

Amazon EKS garantisce che ogni cluster abbia un proprio piano di controllo (control-plane) Kubernetes specifico. Questo design mantiene l'infrastruttura di ciascun cluster separata, senza sovrapposizioni tra cluster o account AWS. La configurazione include:

Componenti distribuiti

Il piano di controllo colloca almeno due istanze del server API e tre istanze `etcd` su tre zone di disponibilità AWS all'interno di una Regione AWS.

Prestazioni ottimali

Amazon EKS monitora e regola attivamente le istanze del piano di controllo per mantenere le massime prestazioni.

Resilienza

Se un'istanza del piano di controllo fallisce, Amazon EKS la sostituisce rapidamente, utilizzando all'occorrenza zone di disponibilità diverse.

Operatività costante

Eseguendo i cluster su più zone di disponibilità, si consegue l'affidabilità dell'Accordo sul livello di servizio (SLA) sulla disponibilità degli endpoint dei server API.

Amazon EKS utilizza Amazon Virtual Private Cloud (Amazon VPC) per limitare il traffico tra i componenti del piano di controllo all'interno di un singolo cluster. I componenti del cluster non sono in grado di visualizzare o ricevere comunicazioni da altri cluster o altri account AWS, ad eccezione di quanto autorizzato con le policy di controllo degli accessi basato sul ruolo (RBAC) Kubernetes.

Calcolo

Oltre al piano di controllo, un cluster Amazon EKS dispone di un set di macchine di lavoro chiamate nodi. La selezione del tipo di nodo del cluster Amazon EKS appropriato è fondamentale per soddisfare i propri requisiti specifici e ottimizzare l'utilizzo delle risorse. Amazon EKS offre i seguenti tipi di nodi primari:

AWS Fargate

Fargate è un motore di elaborazione serverless per container che elimina la necessità di gestire le istanze sottostanti. Con Fargate, è possibile specificare le esigenze di risorse di un'applicazione e AWS esegue automaticamente il provisioning, il dimensionamento e la manutenzione dell'infrastruttura. Questa opzione è ideale per gli utenti che danno priorità alla facilità d'uso e vogliono concentrarsi sullo sviluppo e sull'implementazione delle applicazioni anziché sulla gestione dell'infrastruttura.

Karpenter

Karpenter è una soluzione di dimensionamento automatico dei cluster Kubernetes, flessibile e ad alte prestazioni, che aiuta a migliorare la disponibilità delle applicazioni e l'efficienza del cluster. Karpenter avvia risorse di calcolo della giusta dimensione in risposta alla modifica del carico delle applicazioni in meno di un minuto. Questa opzione consente di fornire risorse di elaborazione just-in-time che soddisfano i requisiti del carico di lavoro.

Gruppi di nodi gestiti

I gruppi di nodi gestiti sono una combinazione di automazione e personalizzazione per la gestione di una raccolta di istanze Amazon EC2 all'interno di un cluster Amazon EKS. AWS si occupa di attività come l'applicazione di patch, l'aggiornamento e il dimensionamento dei nodi, semplificando gli aspetti operativi. Parallelamente, sono supportati gli argomenti `kubelet` personalizzati, che offrono la possibilità di creare policy avanzate per la gestione della CPU e della memoria. Inoltre, la sicurezza viene migliorata tramite ruoli AWS Identity and Access Management (IAM) per gli account di servizio, riducendo al contempo la necessità di autorizzazioni separate per ciascun cluster.

Nodi autogestiti

I nodi autogestiti offrono il pieno controllo sulle proprie istanze Amazon EC2 all'interno di un cluster Amazon EKS. L'utente è responsabile della gestione, del dimensionamento e della manutenzione dei nodi, disponendo del controllo totale sull'infrastruttura sottostante. Questa opzione è consigliata per gli utenti che necessitano di controllo e personalizzazione granulari dei propri nodi e sono pronti a investire tempo nella gestione e nella manutenzione della propria infrastruttura.

Concetti Kubernetes

Amazon Elastic Kubernetes Service (Amazon EKS) AWS è un servizio gestito basato sul progetto open source. Kubernetes Sebbene ci siano cose che devi sapere su come il servizio Amazon EKS si integra con il AWS cloud (in particolare quando crei per la prima volta un cluster Amazon EKS), una volta che è attivo e funzionante, puoi utilizzare il cluster Amazon EKS più o meno allo stesso modo di qualsiasi altro Kubernetes cluster. Quindi, per iniziare a gestire Kubernetes i cluster e distribuire carichi di lavoro, è necessaria almeno una conoscenza di base dei concetti. Kubernetes

Questa pagina divide Kubernetes i concetti in tre sezioni: PerchéKubernetes, Cluster e Carichi di lavoro. La prima sezione descrive il valore dell'esecuzione di un Kubernetes servizio, in particolare

come servizio gestito come Amazon EKS. La sezione Carichi di lavoro spiega come Kubernetes le applicazioni vengono create, archiviate, eseguite e gestite. La sezione Cluster illustra i diversi componenti che compongono i Kubernetes cluster e le responsabilità dell'utente per la creazione e la manutenzione dei cluster. Kubernetes

Argomenti

Durante la lettura di questo contenuto, i link ti condurranno a ulteriori descrizioni dei Kubernetes concetti in Amazon EKS e nella Kubernetes documentazione, nel caso in cui desideri approfondire uno degli argomenti trattati qui. Per dettagli su come Amazon EKS implementa Kubernetes il piano di controllo e le funzionalità di calcolo, consulta l'architettura Amazon EKS.

Perché Kubernetes?

Kubernetesè stato progettato per migliorare la disponibilità e la scalabilità durante l'esecuzione di applicazioni containerizzate di importanza critica e di qualità produttiva. Anziché essere eseguito solo Kubernetes su una singola macchina (sebbene ciò sia possibile), Kubernetes raggiunge questi obiettivi consentendoti di eseguire applicazioni su set di computer che possono espandersi o contrarsi per soddisfare la domanda. Kubernetesinclude funzionalità che semplificano le seguenti operazioni:

- Distribuisci le applicazioni su più macchine (utilizzando contenitori distribuiti nei pod)
- Monitora lo stato dei container e riavvia i container guasti
- Aumenta e riduci i contenitori in base al carico
- Aggiorna i contenitori con nuove versioni
- Alloca le risorse tra i contenitori
- Bilancia il traffico tra le macchine

L'Kubernetesautomazione di questi tipi di attività complesse consente agli sviluppatori di applicazioni di concentrarsi sulla creazione e sul miglioramento dei carichi di lavoro delle applicazioni, anziché preoccuparsi dell'infrastruttura. Lo sviluppatore crea in genere file di configurazione, formattati come file YAML, che descrivono lo stato desiderato dell'applicazione. Ciò potrebbe includere i contenitori

da eseguire, i limiti delle risorse, il numero di repliche dei Pod, l'allocazione della CPU/memoria, le regole di affinità e altro ancora.

Attributi di Kubernetes

Per raggiungere i suoi obiettivi, Kubernetes ha le seguenti caratteristiche:

- Containerizzato: Kubernetes è uno strumento di orchestrazione dei container. Per utilizzarloKubernetes, devi prima avere le tue applicazioni containerizzate. A seconda del tipo di applicazione, potrebbe trattarsi di un insieme di microservizi, di processi in batch o in altre forme. Le applicazioni possono quindi trarre vantaggio da un Kubernetes flusso di lavoro che comprende un enorme ecosistema di strumenti, in cui i contenitori possono essere archiviati come immagini in un registro di contenitori, distribuiti in un Kubernetescluster ed eseguiti su un nodo disponibile. È possibile creare e testare singoli contenitori sul computer locale utilizzando Docker o un altro container runtime, prima di distribuirli nel cluster. Kubernetes
- Scalabile: se la richiesta delle applicazioni supera la capacità delle istanze in esecuzione di tali applicazioni, Kubernetes è in grado di scalare verso l'alto. Se necessario, Kubernetes può stabilire se le applicazioni richiedono più CPU o memoria e rispondere espandendo automaticamente la capacità disponibile o utilizzando una maggiore capacità esistente. La scalabilità può essere eseguita a livello di Pod, se è disponibile una quantità di calcolo sufficiente per eseguire solo più istanze dell'applicazione (scalabilità automatica orizzontale dei Pod), oppure a livello di nodo, se è necessario attivare più nodi per gestire l'aumento della capacità (Cluster Autoscaler o Karpenter). Poiché la capacità non è più necessaria, questi servizi possono eliminare i Pod non necessari e chiudere i nodi non necessari.
- Disponibile: se un'applicazione o un nodo non sono integri o non sono disponibili, Kubernetes possono spostare i carichi di lavoro in esecuzione su un altro nodo disponibile. Puoi forzare il problema semplicemente eliminando un'istanza in esecuzione di un carico di lavoro o di un nodo che esegue i tuoi carichi di lavoro. La conclusione è che i carichi di lavoro possono essere trasferiti in altre posizioni se non possono più essere eseguiti dove si trovano.
- Dichiarativo: Kubernetes utilizza la riconciliazione attiva per verificare costantemente che lo stato dichiarato per il cluster corrisponda allo stato effettivo. Applicando Kubernetesoggetti a un cluster, in genere tramite file di configurazione in formato YAML, puoi, ad esempio, chiedere di avviare i carichi di lavoro che desideri eseguire sul cluster. Successivamente puoi modificare le configurazioni per fare qualcosa come utilizzare una versione successiva di un contenitore o allocare più memoria. Kubernetesfarà ciò che deve fare per stabilire lo stato desiderato. Ciò può includere l'attivazione o la disattivazione dei nodi, l'arresto e il riavvio dei carichi di lavoro o l'estrazione di contenitori aggiornati.

- Componibile: poiché un'applicazione è in genere composta da più componenti, è necessario essere in grado di gestire insieme un insieme di questi componenti (spesso rappresentati da più contenitori). Sebbene Docker Compose offra un modo per farlo direttamente conDocker, il comando Kubernetes Kompose può aiutarti a farlo con. Kubernetes Vedi Translate a Docker Compose File to Kubernetes Resources per un esempio di come eseguire questa operazione.
- Estensibile: a differenza del software proprietario, il Kubernetes progetto open source è progettato per essere aperto all'utente e può essere esteso in Kubernetes qualsiasi modo si desideri per soddisfare le proprie esigenze. Le API e i file di configurazione sono aperti a modifiche dirette. Le terze parti sono incoraggiate a scrivere i propri controller, per estendere sia l'infrastruttura che le funzionalità dell'utente finale. Kubernetes I webhook consentono di configurare le regole dei cluster per applicare le politiche e adattarsi alle mutevoli condizioni. Per altre idee su come estendere i Kubernetes cluster, consulta Extending. Kubernetes
- Portatile: molte organizzazioni hanno standardizzato le proprie operazioni in Kubernetes quanto ciò consente loro di gestire tutte le esigenze applicative nello stesso modo. Gli sviluppatori possono utilizzare le stesse pipeline per creare e archiviare applicazioni containerizzate. Queste applicazioni possono quindi essere distribuite su Kubernetes cluster in esecuzione in locale, nel cloud, sui point-of-sales terminali dei ristoranti o su dispositivi IOT distribuiti tra i siti remoti dell'azienda. La sua natura open source consente alle persone di sviluppare queste Kubernetes distribuzioni speciali, insieme agli strumenti necessari per gestirle.

Gestione di Kubernetes

Kubernetesil codice sorgente è disponibile gratuitamente, quindi con le proprie apparecchiature è possibile installarlo e gestirlo Kubernetes autonomamente. Tuttavia, l'autogestione Kubernetes richiede una profonda esperienza operativa e richiede tempo e impegno per la manutenzione. Per questi motivi, la maggior parte delle persone che implementano carichi di lavoro di produzione sceglie un provider cloud (come Amazon EKS) o un provider locale (come Amazon EKS Anywhere) con la propria Kubernetes distribuzione testata e il supporto di esperti. Kubernetes Ciò consente di alleggerire gran parte del carico di lavoro indifferenziato necessario per la manutenzione dei cluster, tra cui:

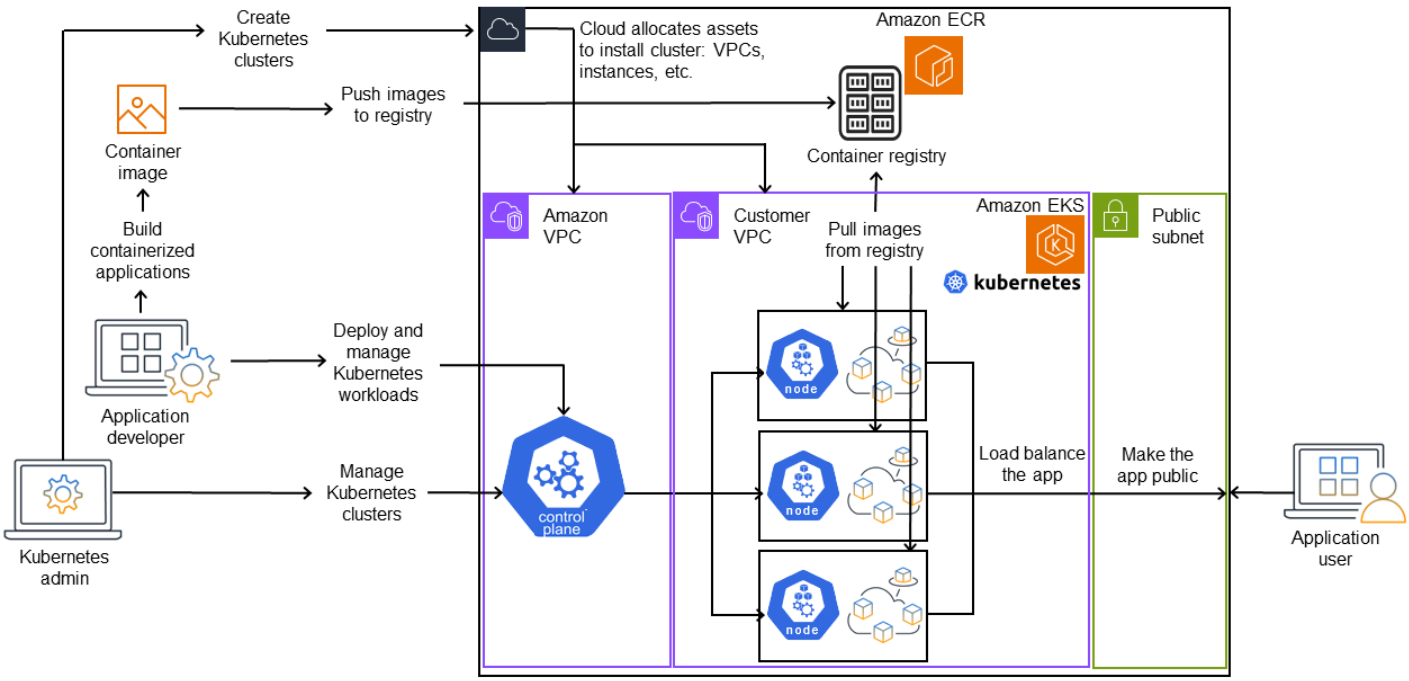
- Hardware: se non disponi di hardware disponibile Kubernetes per l'esecuzione in base alle tue esigenze, un provider di servizi cloud come AWS Amazon EKS può farti risparmiare sui costi iniziali. Con Amazon EKS, ciò significa che puoi utilizzare le migliori risorse cloud offerte da AWS, tra cui istanze di computer (Amazon Elastic Compute Cloud), il tuo ambiente privato (Amazon VPC), la gestione centralizzata di identità e permessi (IAM) e lo storage (Amazon EBS). AWS gestisce i computer, le reti, i data center e tutti gli altri componenti fisici necessari per l'esecuzione.

Kubernetes Allo stesso modo, non è necessario pianificare il data center per gestire la capacità massima nei giorni di maggiore richiesta. Per Amazon EKS Anywhere o altri Kubernetes cluster locali, sei responsabile della gestione dell'infrastruttura utilizzata nelle Kubernetes distribuzioni, ma puoi comunque contare su di te AWS per tenerti Kubernetes aggiornato.

- Gestione del piano di controllo: Amazon EKS gestisce la sicurezza e la disponibilità del piano di Kubernetes controllo AWS ospitato, che è responsabile della pianificazione dei container, della gestione della disponibilità delle applicazioni e di altre attività chiave, in modo che tu possa concentrarti sui carichi di lavoro delle applicazioni. In caso di guasto del cluster, AWS dovresti disporre dei mezzi per ripristinarlo in uno stato di esecuzione. Per Amazon EKS Anywhere, gestiresti tu stesso il piano di controllo.
- Upgrade testati: quando aggiorni i tuoi cluster, puoi fare affidamento su Amazon EKS o Amazon EKS Anywhere per fornire versioni testate delle loro Kubernetes distribuzioni.
- Componenti aggiuntivi: esistono centinaia di progetti progettati per estenderli e utilizzarli, Kubernetes che puoi aggiungere all'infrastruttura del cluster o utilizzare per facilitare l'esecuzione dei tuoi carichi di lavoro. Invece di creare e gestire autonomamente questi componenti aggiuntivi, AWS fornisce componenti aggiuntivi Amazon EKS che puoi usare con i tuoi cluster. Amazon EKS Anywhere fornisce pacchetti curati che includono build di molti progetti open source popolari. Quindi non è necessario creare il software da soli o gestire patch di sicurezza, correzioni di bug o aggiornamenti critici. Allo stesso modo, se le impostazioni predefinite soddisfano le vostre esigenze, è normale che sia necessaria una configurazione minima di tali componenti aggiuntivi. Vedi Extend Clusters per i dettagli sull'estensione del cluster con componenti aggiuntivi.

Kubernetesin azione

Il diagramma seguente mostra le attività chiave che potresti svolgere come Kubernetes amministratore o sviluppatore di applicazioni per creare e utilizzare un Kubernetes cluster. Nel processo, illustra come i Kubernetes componenti interagiscono tra loro, utilizzando il AWS cloud come esempio del provider cloud sottostante.



Un Kubernetes amministratore crea il Kubernetes cluster utilizzando uno strumento specifico per il tipo di provider su cui verrà creato il cluster. Questo esempio utilizza il AWS cloud come provider, che offre il Kubernetes servizio gestito chiamato Amazon EKS. Il servizio gestito alloca automaticamente le risorse necessarie per creare il cluster, inclusa la creazione di due nuovi cloud privati virtuali (Amazon VPC) per il cluster, la configurazione della rete, la mappatura Kubernetes delle autorizzazioni per la gestione degli asset nel cloud, la verifica che i servizi del piano di controllo abbiano aree in cui eseguire e l'allocazione di zero o più istanze Amazon EC2 come nodi per l'esecuzione dei carichi di lavoro. Kubernetes AWS gestisce un Amazon VPC stesso per il piano di controllo, mentre l'altro Amazon VPC contiene i nodi cliente che eseguono i carichi di lavoro.

Molte delle attività future dell'Kubernetesamministratore vengono eseguite utilizzando Kubernetes strumenti come kubectl. Questo strumento invia richieste di servizi direttamente al piano di controllo del cluster. I modi in cui le interrogazioni e le modifiche vengono apportate al cluster sono quindi molto simili a quelli in cui le eseguiresti su qualsiasi Kubernetes cluster.

Uno sviluppatore di applicazioni che desidera distribuire carichi di lavoro in questo cluster può eseguire diverse attività. Lo sviluppatore deve creare l'applicazione in una o più immagini di container, quindi inviarle a un registro di container accessibile al cluster. Kubernetes AWS a tale

scopo offre Amazon Elastic Container Registry (Amazon ECR) Elastic Container Registry (Amazon ECR).

Per eseguire l'applicazione, lo sviluppatore può creare file di configurazione in formato YAML che indicano al cluster come eseguire l'applicazione, compresi quali contenitori estrarre dal registro e come avvolgerli in Pods. Il piano di controllo (scheduler) pianifica i contenitori su uno o più nodi e il runtime del contenitore su ciascun nodo recupera ed esegue effettivamente i contenitori necessari. Lo sviluppatore può anche configurare un sistema di bilanciamento del carico delle applicazioni per bilanciare il traffico verso i container disponibili in esecuzione su ciascun nodo ed esporre l'applicazione al mondo esterno in modo che sia disponibile su una rete pubblica. Fatto ciò, qualcuno che desidera utilizzare l'applicazione può connettersi all'endpoint dell'applicazione per accedervi.

La sezione seguente illustra i dettagli di ciascuna di queste funzionalità, dal punto di vista dei Kubernetes cluster e dei carichi di lavoro.

Cluster

Se il tuo compito è avviare e gestire Kubernetes i cluster, dovresti sapere come i Kubernetes cluster vengono creati, migliorati, gestiti ed eliminati. È inoltre necessario sapere quali sono i componenti che costituiscono un cluster e cosa è necessario fare per mantenerli.

Gli strumenti per la gestione dei cluster gestiscono la sovrapposizione tra i Kubernetes servizi e il provider hardware sottostante. Per questo motivo, l'automazione di queste attività tende ad essere eseguita dal Kubernetes provider (come Amazon EKS o Amazon EKS Anywhere) utilizzando strumenti specifici per il provider. Ad esempio, puoi usare per avviare un cluster Amazon EKS`eksctl create cluster`, mentre per Amazon EKS Anywhere puoi usarlo`eksctl anywhere create cluster`. Tieni presente che, sebbene questi comandi creino un Kubernetes cluster, sono specifici del provider e non fanno parte del Kubernetes progetto stesso.

Strumenti per la creazione e la gestione dei cluster

Il Kubernetes progetto offre strumenti per la creazione manuale di un Kubernetes cluster. Quindi, se desideri eseguire l'installazione Kubernetes su una singola macchina o eseguire il piano di controllo su una macchina e aggiungere nodi manualmente, puoi utilizzare strumenti CLI come kind, minikube o kubeadm elencati in Strumenti di installazione. Kubernetes Per semplificare e automatizzare l'intero ciclo di vita della creazione e della gestione dei cluster, è molto più semplice utilizzare strumenti supportati da un Kubernetes provider affermato, come Amazon EKS o Amazon EKS Anywhere.

Nel AWS cloud, puoi creare cluster Amazon EKS utilizzando strumenti CLI, come eksctl, o strumenti più dichiarativi, come Terraform (vedi Amazon EKS Blueprints for Terraform). Puoi anche creare

un cluster dalla console di gestione. AWS Consulta le funzionalità di Amazon EKS per un elenco di ciò che ottieni con Amazon EKS. Kubernetesle responsabilità che Amazon EKS si assume per te includono:

- Piano di controllo gestito: AWS assicura che il cluster Amazon EKS sia disponibile e scalabile perché gestisce il piano di controllo per te e lo rende disponibile in tutte le zone di AWS disponibilità.
- Gestione dei nodi: invece di aggiungere nodi manualmente, puoi fare in modo che Amazon EKS crei i nodi automaticamente secondo necessità, utilizzando Managed Node Groups o Karpenter. I gruppi di nodi gestiti hanno integrazioni con Kubernetes Cluster Autoscaling. Utilizzando gli strumenti di gestione dei nodi, puoi sfruttare i risparmi sui costi, ad esempio le istanze Spot e il consolidamento dei nodi, e la disponibilità, utilizzando le funzionalità di pianificazione per impostare la modalità di distribuzione dei carichi di lavoro e la selezione dei nodi.
- Rete in cluster: utilizzando CloudFormation modelli, `eksctl` configura la rete tra i componenti del piano di controllo e del piano dati (nodo) nel cluster. Kubernetes Configura inoltre gli endpoint attraverso i quali possono avvenire le comunicazioni interne ed esterne. Per ulteriori dettagli, consulta Demistificare la rete di cluster per i nodi di lavoro Amazon EKS. Le comunicazioni tra i Pod in Amazon EKS vengono effettuate utilizzando Amazon EKS Pod Identities, che consente ai Pod di attingere ai metodi AWS cloud per la gestione di credenziali e autorizzazioni.
- Componenti aggiuntivi: Amazon EKS ti evita di dover creare e aggiungere componenti software comunemente usati per supportare i Kubernetes cluster. Ad esempio, quando crei un cluster Amazon EKS dalla console di AWS gestione, vengono aggiunti automaticamente il kube-proxy Amazon EKS, il plug-in Amazon VPC CNI per e i componenti aggiuntivi CoredNS. Kubernetes Consulta i componenti aggiuntivi di Amazon EKS per ulteriori informazioni su questi componenti aggiuntivi, incluso un elenco dei componenti aggiuntivi disponibili.

Per eseguire i cluster su computer e reti locali, Amazon offre Amazon EKS Anywhere. Invece che il AWS cloud sia il provider, puoi scegliere di eseguire Amazon EKS Anywhere su piattaforme VMware vSphere, bare metal (provider Tinkerbell) CloudStack, Snow o Nutanix utilizzando le tue apparecchiature.

Amazon EKS Anywhere si basa sullo stesso software Amazon EKS Distro utilizzato da Amazon EKS. Tuttavia, Amazon EKS Anywhere si affida a diverse implementazioni dell'interfaccia KubernetesCluster API (CAPI) per gestire l'intero ciclo di vita delle macchine in un cluster Amazon EKS Anywhere (come CAPV for vSphere e CAPC for). CloudStack Poiché l'intero cluster è in esecuzione sulle tue apparecchiature, ti assumi la responsabilità aggiuntiva della gestione del piano di controllo e del backup dei dati (vedi etcd più avanti in questo documento).

Componenti del cluster

Kubernetesi componenti del cluster sono suddivisi in due aree principali: piano di controllo e nodi di lavoro. I componenti Control Plane gestiscono il cluster e forniscono l'accesso alle relative API. I nodi di lavoro (a volte denominati semplicemente nodi) forniscono i luoghi in cui vengono eseguiti i carichi di lavoro effettivi. I componenti del nodo sono costituiti da servizi eseguiti su ciascun nodo per comunicare con il piano di controllo ed eseguire i contenitori. Il set di nodi di lavoro per il cluster è denominato Data Plane.

Piano di controllo (control-plane)

Il piano di controllo è costituito da un insieme di servizi che gestiscono il cluster. Questi servizi possono essere eseguiti tutti su un singolo computer o possono essere distribuiti su più computer. Internamente, queste sono denominate Control Plane Instances (CPI). Il modo in cui vengono eseguite le CPI dipende dalla dimensione del cluster e dai requisiti di elevata disponibilità. Con l'aumento della domanda nel cluster, un servizio del piano di controllo può scalare per fornire più istanze di quel servizio, con il bilanciamento del carico delle richieste tra le istanze.

Le attività eseguite dai componenti del piano di Kubernetes controllo includono:

- Comunicazione con i componenti del cluster (server API): il server API (kube-apiserver) espone l'KubernetesAPI in modo che le richieste al cluster possano essere effettuate sia dall'interno che dall'esterno del cluster. In altre parole, le richieste di aggiunta o modifica degli oggetti di un cluster (Pods, Services, Nodes e così via) possono provenire da comandi esterni, come le richieste di esecuzione di un Pod. `kubectl` Allo stesso modo, è possibile effettuare richieste dal server API ai componenti all'interno del cluster, ad esempio una richiesta al `kubelet` servizio per lo stato di un Pod.
- Archivia dati sul cluster (etcd key value store) - Il servizio etcd svolge il ruolo fondamentale di tenere traccia dello stato corrente del cluster. Se il servizio etcd diventasse inaccessibile, non sarebbe possibile aggiornare o interrogare lo stato del cluster, anche se i carichi di lavoro continuerebbero a funzionare per un po'. Per questo motivo, i cluster critici in genere dispongono di più istanze del servizio etcd con bilanciamento del carico in esecuzione contemporaneamente ed eseguono backup periodici dell'archivio di valori della chiave etcd in caso di perdita o danneggiamento dei dati. Tieni presente che, in Amazon EKS, tutto questo viene gestito automaticamente per impostazione predefinita. Amazon EKS Anywhere fornisce istruzioni per il backup e il ripristino di etcd. Consulta il modello di dati etcd per scoprire come etcd gestisce i dati.
- Schedule Pods to nodes (Scheduler) - Le richieste di avvio o arresto di un Pod in Kubernetes vengono indirizzate allo Scheduler (Kuberneteskube-scheduler). Poiché un cluster può avere più

nodi in grado di eseguire il Pod, spetta allo Scheduler scegliere su quale nodo (o nodi, nel caso delle repliche) eseguire il Pod. Se la capacità disponibile non è sufficiente per eseguire il Pod richiesto su un nodo esistente, la richiesta avrà esito negativo, a meno che non siano state prese altre disposizioni. Tali disposizioni potrebbero includere l'abilitazione di servizi come Managed Node Groups o Karpenter in grado di avviare automaticamente nuovi nodi per gestire i carichi di lavoro.

- Mantieni i componenti nello stato desiderato (Controller Manager): Kubernetes Controller Manager viene eseguito come processo daemon (kube-controller-manager) per controllare lo stato del cluster e apportare modifiche al cluster per ristabilire gli stati previsti. In particolare, esistono diversi controller che controllano diversi Kubernetes oggetti, tra cui statefulset-controller, endpoint-controller node-lifecycle-controller, cronjob-controller e altri.
- Gestisci le risorse cloud (Cloud Controller Manager): le interazioni tra Kubernetes e il provider cloud che esegue le richieste per le risorse del data center sottostante sono gestite da Cloud Controller Manager (). cloud-controller-manager I controller gestiti da Cloud Controller Manager possono includere un controller di percorso (per configurare i percorsi di rete cloud), un controller di servizio (per utilizzare i servizi di bilanciamento del carico nel cloud) e un controller di nodi (per utilizzare le API cloud per mantenere Kubernetes i nodi sincronizzati con i nodi cloud).

Worker Nodes (piano dati)

Per un Kubernetes cluster a nodo singolo, i carichi di lavoro vengono eseguiti sulla stessa macchina del piano di controllo. Tuttavia, una configurazione più normale consiste nell'avere uno o più sistemi informatici separati (nodi) dedicati all'esecuzione Kubernetes dei carichi di lavoro.

Quando si crea un Kubernetes cluster per la prima volta, alcuni strumenti per la creazione di cluster consentono di configurare un certo numero di nodi da aggiungere al cluster (identificando i sistemi informatici esistenti o chiedendo al provider di crearne di nuovi). Prima di aggiungere qualsiasi carico di lavoro a tali sistemi, vengono aggiunti servizi a ciascun nodo per implementare queste funzionalità:

- Gestisci ogni nodo (kubelet): il server API comunica con il servizio kubelet in esecuzione su ciascun nodo per assicurarsi che il nodo sia registrato correttamente e che i Pod richiesti dallo Scheduler siano in esecuzione. Il kubelet può leggere i manifesti dei Pod e configurare volumi di archiviazione o altre funzionalità necessarie ai Pod sul sistema locale. Può anche verificare lo stato dei container in esecuzione localmente.
- Esegui contenitori su un nodo (runtime del contenitore): il Container Runtime su ogni nodo gestisce i contenitori richiesti per ogni Pod assegnato al nodo. Ciò significa che può estrarre le immagini del contenitore dal registro appropriato, eseguire il contenitore, interromperlo e rispondere alle

domande sul contenitore. Il runtime predefinito del contenitore è containerd. A partire dalla Kubernetes 1.24, la speciale integrazione di Docker (Dockershim) che poteva essere utilizzata come runtime del contenitore è stata eliminata. Kubernetes Sebbene sia ancora possibile Docker utilizzarlo per testare ed eseguire contenitori sul sistema locale, per utilizzarlo Docker con ora Kubernetes è necessario installare Docker Engine su ciascun nodo con cui utilizzarlo. Kubernetes

- Gestione della rete tra contenitori (kube-proxy): per poter supportare la comunicazione tra i Pod tramite i Servizi, era Kubernetes necessario un modo per configurare le reti Pod per tenere traccia degli indirizzi IP e delle porte associate a tali Pod. Il servizio kube-proxy viene eseguito su ogni nodo per consentire la comunicazione tra i Pod.

Estendi i cluster

È possibile aggiungere alcuni servizi per Kubernetes supportare il cluster, ma non vengono eseguiti nel piano di controllo. Questi servizi spesso vengono eseguiti direttamente sui nodi dello spazio dei nomi del sistema kube o nel relativo spazio dei nomi (come spesso accade con fornitori di servizi di terze parti). Un esempio comune è il servizio CoredNS, che fornisce servizi DNS al cluster. Fai riferimento a Discovering builtin services per informazioni su come vedere quali servizi cluster sono in esecuzione nel sistema kube sul tuo cluster.

Esistono diversi tipi di componenti aggiuntivi che puoi considerare di aggiungere ai tuoi cluster. Per mantenere integri i cluster, puoi aggiungere funzionalità di osservabilità che ti consentono di eseguire operazioni come la registrazione, il controllo e le metriche. Con queste informazioni, puoi risolvere i problemi che si verificano, spesso tramite le stesse interfacce di osservabilità. Esempi di questi tipi di servizi includono Amazon GuardDuty, AWS Distro for CloudWatchOpenTelemetry, il plug-in Amazon VPC CNI per e Grafana Kubernetes Monitoring. Kubernetes Per lo storage, i componenti aggiuntivi di Amazon EKS includono Amazon Elastic Block Store CSI Driver (per aggiungere dispositivi di storage a blocchi), Amazon Elastic File System CSI Driver (per aggiungere storage di file system) e diversi componenti aggiuntivi di storage di terze parti (come Amazon FSx per il driver CSI ONTAP). NetApp

Per un elenco più completo dei componenti aggiuntivi Amazon EKS disponibili, consulta i componenti aggiuntivi di Amazon EKS.

Carichi di lavoro

Kubernetesdefinisce un carico di lavoro come «un'applicazione in esecuzione». Kubernetes Tale applicazione può essere costituita da un set di microservizi eseguiti come Containers in Pods oppure può essere eseguita come processo batch o altro tipo di applicazioni. Il compito di Kubernetes è

assicurarsi che le richieste effettuate per la configurazione o la distribuzione di tali oggetti vengano eseguite. Chi distribuisce applicazioni dovrebbe imparare come vengono creati i container, come vengono definiti i Pod e quali metodi è possibile utilizzare per distribuirli.

Container

L'elemento più basilare del carico di lavoro di un'applicazione in cui distribuire e gestire è un Pod. Kubernetes Un Pod rappresenta un modo per contenere i componenti di un'applicazione e per definire le specifiche che descrivono gli attributi del Pod. Confrontalo con qualcosa come un pacchetto RPM o Deb, che raggruppa il software per un sistema Linux, ma non funziona di per sé come un'entità.

Poiché il Pod è l'unità dispiegabile più piccola, in genere contiene un singolo contenitore. Tuttavia, in un Pod possono essere presenti più contenitori nei casi in cui i contenitori siano strettamente collegati. Ad esempio, un contenitore di server Web potrebbe essere impacchettato in un Pod con un tipo di contenitore secondario che può fornire la registrazione, il monitoraggio o altri servizi strettamente legati al contenitore del server Web. In questo caso, la presenza nello stesso Pod garantisce che per ogni istanza in esecuzione del Pod, entrambi i contenitori vengano sempre eseguiti sullo stesso nodo. Allo stesso modo, tutti i contenitori in un Pod condividono lo stesso ambiente, con i contenitori in un Pod che funzionano come se si trovassero nello stesso host isolato. L'effetto di ciò è che i contenitori condividono un unico indirizzo IP che fornisce l'accesso al Pod e possono comunicare tra loro come se fossero in esecuzione sul proprio localhost.

Le specifiche del Pod (PodSpec) definiscono lo stato desiderato del Pod. Puoi distribuire un singolo Pod o più Pod utilizzando le risorse del carico di lavoro per gestire i modelli Pod. Le risorse per il carico di lavoro includono le implementazioni (per gestire più repliche di pod), StatefulSets(per distribuire pod che devono essere unici, come i pod del database) e DaemonSets(dove un pod deve funzionare continuamente su ogni nodo). Ne parleremo più avanti.

Sebbene un Pod sia l'unità più piccola che installi, un container è l'unità più piccola che costruisci e gestisci.

Contenitori da costruzione

Il Pod è in realtà solo una struttura attorno a uno o più contenitori, con ogni contenitore stesso che contiene il file system, gli eseguibili, i file di configurazione, le librerie e altri componenti per eseguire effettivamente l'applicazione. Poiché una società chiamata Docker Inc. ha reso popolari i contenitori per la prima volta, alcune persone li chiamano Containers. Docker Tuttavia, da allora l'Open Container Initiative ha definito i tempi di esecuzione, le immagini e i metodi di distribuzione

dei container per il settore. A ciò si aggiunge il fatto che i container sono stati creati partendo da molte funzionalità Linux esistenti, altri spesso si riferiscono ai contenitori come OCI Containers, Linux Containers o semplicemente Containers.

Quando si crea un contenitore, in genere si inizia con un Docker file (chiamato letteralmente così). All'interno di quel Dockerfile, identifichi:

- Un'immagine di base - Un'immagine contenitore di base è un contenitore che viene in genere creato da una versione minima del file system di un sistema operativo (come Red Hat Enterprise Linux o Ubuntu) o da un sistema minimale migliorato per fornire software per eseguire tipi specifici di applicazioni (come nodejs o app python).
- Software applicativo: puoi aggiungere il tuo software applicativo al tuo contenitore più o meno nello stesso modo in cui lo aggiungeresti a un sistema Linux. Ad esempio, nel tuo Dockerfile puoi eseguire `npm` e `yarn` installare un'applicazione Java o `dnf` installare `yum` pacchetti RPM. In altre parole, utilizzando un comando RUN in un Dockerfile, è possibile eseguire qualsiasi comando disponibile nel file system dell'immagine di base per installare software o configurare software all'interno dell'immagine contenitore risultante.
- Istruzioni - Il riferimento a Dockerfile descrive le istruzioni che è possibile aggiungere a un Dockerfile quando lo si configura. Queste includono le istruzioni utilizzate per creare ciò che è contenuto nel contenitore stesso (`ADD`o nei `COPY` file dal sistema locale), identificare i comandi da eseguire quando il contenitore viene eseguito (`CMD`o `ENTRYPOINT`) e connettere il contenitore al sistema su cui viene eseguito (identificando il contenitore da `USER` eseguire, un locale da montare o le `VOLUME` porte su cui eseguire). `EXPOSE`

Sebbene il `docker` comando e il servizio siano stati tradizionalmente utilizzati per creare contenitori (`docker build`), altri strumenti disponibili per creare immagini di contenitori includono podman e nerdctl. Vedi Building Better Container Images o Build with Docker per ulteriori informazioni sulla creazione di contenitori.

Conservazione dei contenitori

Dopo aver creato l'immagine del contenitore, puoi archiviarla in un registro di distribuzione dei container sulla tua workstation o in un registro pubblico dei container. L'esecuzione di un registro privato dei contenitori sulla workstation consente di archiviare le immagini dei contenitori localmente, rendendole immediatamente disponibili.

Per archiviare le immagini dei contenitori in modo più pubblico, puoi inviarle a un registro di contenitori pubblico. I registri pubblici dei container forniscono una posizione centrale per

l'archiviazione e la distribuzione delle immagini dei container. Esempi di registri di container pubblici includono Amazon Elastic Container Registry, Red Hat Quay registry e Docker Hub registry.

Quando esegui carichi di lavoro containerizzati su Amazon Elastic Kubernetes Service (Amazon EKS), consigliamo di Docker estrarre copie delle immagini ufficiali archiviate in Amazon Elastic Container Registry. AWS Amazon ECR archivia queste immagini dal 2021. Puoi cercare le immagini dei container più comuni nella Galleria pubblica di Amazon ECR e, in particolare, per le immagini dell'DockerHub, puoi cercare nella Galleria Amazon ECR. Docker

Contenitori in esecuzione

Poiché i contenitori sono creati in un formato standard, un contenitore può essere eseguito su qualsiasi macchina in grado di eseguire un runtime di container (ad esempioDocker) e il cui contenuto corrisponda all'architettura della macchina locale (ad esempio `x86_64` o`arm`). Per testare un contenitore o semplicemente eseguirlo sul desktop locale, puoi utilizzare `podman run` i comandi `docker run` or per avviare un contenitore sul localhost. TuttaviaKubernetes, ogni nodo di lavoro ha un container runtime distribuito ed è compito di richiedere che un nodo Kubernetes esegua un container.

Una volta che un contenitore è stato assegnato per l'esecuzione su un nodo, il nodo verifica se la versione richiesta dell'immagine del contenitore esiste già sul nodo. In caso contrario, Kubernetes indica al runtime del contenitore di estrarre quel contenitore dal registro dei contenitori appropriato, quindi di eseguirlo localmente. Tieni presente che l'immagine del contenitore si riferisce al pacchetto software che viene spostato tra il laptop, il registro del contenitore e Kubernetes i nodi. Un contenitore si riferisce a un'istanza in esecuzione di quell'immagine.

Cialde

Una volta che i contenitori sono pronti, l'utilizzo dei pod include la configurazione, la distribuzione e l'accessibilità dei pod.

Configurazione dei pod

Quando definisci un Pod, gli assegni una serie di attributi. Questi attributi devono includere almeno il nome del Pod e l'immagine del contenitore da eseguire. Tuttavia, ci sono anche molte altre cose che vuoi configurare con le definizioni dei tuoi Pod (consulta la PodSecpagina per i dettagli su cosa può essere inserito in un Pod). Ciò include:

- Archiviazione: quando un contenitore in esecuzione viene interrotto ed eliminato, l'archiviazione dei dati in quel contenitore scompare, a meno che non si configuri uno spazio di archiviazione più

permanente. Kubernetessupporta molti tipi di storage diversi e li riassume sotto l'egida di Volumes. I tipi di storage includono CephFS, NFS, iSCSI e altri. È anche possibile utilizzare un dispositivo a blocchi locale dal computer locale. Con uno di questi tipi di archiviazione disponibili nel cluster, è possibile montare il volume di archiviazione su un punto di montaggio selezionato nel file system del contenitore. Un volume persistente è quello che continua a esistere dopo l'eliminazione del Pod, mentre un volume effimero viene eliminato quando il Pod viene eliminato. Se l'amministratore del cluster ne ha creati diversi StorageClassesper il cluster, potreste avere la possibilità di scegliere gli attributi dello storage da utilizzare, ad esempio se il volume viene eliminato o recuperato dopo l'uso, se si espanderà se è necessario più spazio e anche se soddisfa determinati requisiti di prestazioni.

- Segreti: rendendo disponibili Secrets ai contenitori nelle specifiche di Pod, puoi fornire le autorizzazioni necessarie a tali contenitori per accedere a file system, database o altre risorse protette. Chiavi, password e token sono tra gli elementi che possono essere archiviati come segreti. L'uso dei segreti consente di non dover memorizzare queste informazioni nelle immagini dei contenitori, ma è sufficiente rendere i segreti disponibili ai contenitori in esecuzione. Simili a Secrets lo sono ConfigMaps. A ConfigMap tende a contenere informazioni meno critiche, come coppie chiave-valore per la configurazione di un servizio.
- Risorse del contenitore: gli oggetti per un'ulteriore configurazione dei contenitori possono assumere la forma di configurazione delle risorse. Per ogni contenitore, puoi richiedere la quantità di memoria e CPU che può utilizzare, nonché porre limiti alla quantità totale di tali risorse che il contenitore può utilizzare. Vedi Gestione delle risorse per pod e contenitori per esempi.
- Interruzioni: i pod possono essere interrotti involontariamente (un nodo non funziona) o volontariamente (è necessario un aggiornamento). Configurando un budget per le interruzioni dei Pod, puoi esercitare un certo controllo sulla disponibilità dell'applicazione in caso di interruzioni. Per alcuni esempi, vedi Specificazione di un budget per le interruzioni per la tua applicazione.
- Namespace: Kubernetes offre diversi modi per isolare Kubernetes componenti e carichi di lavoro l'uno dall'altro. L'esecuzione di tutti i Pod per una particolare applicazione nello stesso namespace è un modo comune per proteggere e gestire tali Pod insieme. È possibile creare spazi dei nomi personalizzati da utilizzare o scegliere di non indicare uno spazio dei nomi (il che comporta l'utilizzo dello spazio dei nomi). Kubernetes `default` Kubernetesi componenti del piano di controllo in genere vengono eseguiti nello spazio dei nomi del sistema kube.

La configurazione appena descritta viene in genere raccolta in un file YAML da applicare al cluster. Kubernetes Per Kubernetes i cluster personali, è sufficiente archiviare questi file YAML sul sistema locale. Tuttavia, con cluster e carichi di lavoro più critici, GitOpsè un modo popolare

per automatizzare lo storage e gli aggiornamenti sia del carico di lavoro che delle risorse dell'infrastruttura. Kubernetes

Gli oggetti utilizzati per raccogliere e distribuire le informazioni sui Pod sono definiti da uno dei seguenti metodi di distribuzione.

Implementazione di pod

Il metodo da scegliere per distribuire i Pod dipende dal tipo di applicazione che intendi eseguire con tali Pod. Ecco alcune delle tue scelte:

- Applicazioni stateless: un'applicazione stateless non salva i dati della sessione di un client, quindi un'altra sessione non deve fare riferimento a ciò che è accaduto a una sessione precedente. In questo modo è più semplice sostituire i Pod con altri nuovi se non funzionano bene o spostarli senza salvarne lo stato. Se stai eseguendo un'applicazione stateless (come un server web), puoi utilizzare un Deployment per distribuire Pods e. ReplicaSets A ReplicaSet definisce quante istanze di un Pod vuoi che vengano eseguite contemporaneamente. Sebbene sia possibile eseguire ReplicaSet direttamente un Pod, è normale eseguire le repliche direttamente all'interno di un Deployment, per definire quante repliche di un Pod devono essere eseguite contemporaneamente.
- Applicazioni con stato: un'applicazione con stato è un'applicazione in cui l'identità del Pod e l'ordine in cui i Pod vengono avviati sono importanti. Queste applicazioni necessitano di uno storage persistente che sia stabile e che debba essere distribuito e scalato in modo coerente. Per distribuire un'applicazione stateful inKubernetes, puoi usare. StatefulSets Un esempio di applicazione che in genere viene eseguita come un database StatefulSet . All'interno di a StatefulSet, è possibile definire le repliche, il Pod e i relativi contenitori, i volumi di archiviazione da montare e le posizioni nel contenitore in cui vengono archiviati i dati. Vedi Run a Replicated Stateful Application per un esempio di database distribuito come. ReplicaSet
- Applicazioni per nodo: a volte è necessario eseguire un'applicazione su ogni nodo del cluster. Kubernetes Ad esempio, il centro dati potrebbe richiedere che ogni computer esegua un'applicazione di monitoraggio o un particolare servizio di accesso remoto. Ad Kubernetes esempio, è possibile utilizzare DaemonSeta per garantire che l'applicazione selezionata venga eseguita su ogni nodo del cluster.
- Le applicazioni vengono eseguite fino al completamento: ci sono alcune applicazioni che si desidera eseguire per completare una determinata attività. Ciò potrebbe includere uno che esegue report mensili sullo stato o elimina i vecchi dati. Un oggetto Job può essere utilizzato per configurare un'applicazione per l'avvio e l'esecuzione, quindi uscire al termine dell'attività. Un CronJoboggetto consente di configurare un'applicazione in modo che venga eseguita a un'ora,

un minuto, un giorno del mese, del mese o del giorno della settimana specifici, utilizzando una struttura definita dal formato crontab di Linux.

Rendere le applicazioni accessibili dalla rete

Poiché le applicazioni venivano spesso distribuite come un insieme di microservizi che si spostavano in luoghi diversi, Kubernetes era necessario trovare un modo per consentire a tali microservizi di trovarsi l'un l'altro. Inoltre, per consentire ad altri utenti di accedere a un'applicazione esterna al Kubernetes cluster, Kubernetes era necessario un modo per esporre tale applicazione su indirizzi e porte esterni. Queste funzionalità relative alla rete vengono eseguite rispettivamente con oggetti Service e Ingress:

- Servizi - Poiché un Pod può spostarsi su nodi e indirizzi diversi, un altro Pod che deve comunicare con il primo Pod potrebbe avere difficoltà a trovare dove si trova. Per risolvere questo problema, Kubernetes consente di rappresentare un'applicazione come servizio. Con un servizio, è possibile identificare un Pod o un set di Pod con un nome particolare, quindi indicare quale porta espone il servizio di quell'applicazione dal Pod e quali porte potrebbe utilizzare un'altra applicazione per contattare quel servizio. Un altro Pod all'interno di un cluster può semplicemente richiedere un servizio per nome e Kubernetes indirizzare la richiesta alla porta appropriata per un'istanza del Pod che esegue quel servizio.
- Ingress - Ingress è ciò che può rendere disponibili le applicazioni rappresentate dai Kubernetes Servizi ai client che si trovano all'esterno del cluster. Le funzionalità di base di Ingress includono un sistema di bilanciamento del carico (gestito da Ingress), il controller Ingress e regole per il routing delle richieste dal controller al Servizio. Esistono diversi controller di ingresso tra cui scegliere. Kubernetes

Passaggi successivi

Comprendere Kubernetes i concetti di base e il modo in cui si riferiscono ad Amazon EKS ti aiuterà a navigare sia nella documentazione che nella Kubernetesdocumentazione di Amazon EKS per trovare le informazioni necessarie per gestire i cluster Amazon EKS e distribuire carichi di lavoro in tali cluster. Per iniziare a utilizzare Amazon EKS, scegli tra le seguenti opzioni:

- Crea un cluster semplice
- Crea un cluster più complesso
- Implementa un'applicazione di esempio
- Esplora i modi per gestire il tuo cluster

Opzioni di implementazione

È possibile utilizzare Amazon EKS con una o con tutte le seguenti opzioni di implementazione:

Amazon EKS nel cloud

: È possibile eseguire Kubernetes nel cloud AWS senza la necessità di installare, utilizzare e mantenere un proprio piano di controllo (control-plane) o nodi Kubernetes. Questa è l'opzione trattata in questa guida.

Amazon EKS su Outposts

: AWS Outposts abilita infrastrutture, modelli operativi e Servizi AWS nativi nelle strutture on-premise. Con Amazon EKS su Outposts è possibile scegliere se eseguire cluster estesi o locali. Con i cluster estesi, il piano di controllo (control-plane) Kubernetes viene eseguito in una Regione AWS e i nodi vengono eseguiti su un Outpost. Con i cluster locali, l'intero cluster Kubernetes viene eseguito localmente su Outposts, inclusi il piano di controllo (control-plane) e i nodi Kubernetes. Per ulteriori informazioni, consulta Amazon EKS su AWS Outposts.

Amazon EKS Anywhere

: Amazon EKS Anywhere è un'opzione di implementazione per Amazon EKS che consente di creare e gestire facilmente cluster Kubernetes on-premise. Sia Amazon EKS che Amazon EKS Anywhere sono costruiti su Amazon EKS Distro. Per maggiori informazioni su Amazon EKS Anywhere e sulle sue differenze con Amazon EKS, consulta Panoramica e Confronto di Amazon EKS Anywhere con Amazon EKS nella documentazione di Amazon EKS Anywhere. Per le risposte ad alcune domande comuni, consulta la pagina Domande frequenti su Amazon EKS Anywhere.

Amazon EKS Distro

: Amazon EKS Distro è una distribuzione dello stesso software Kubernetes open source e delle dipendenze distribuite da Amazon EKS nel cloud. Amazon EKS Distro segue lo stesso ciclo di rilascio della versione Kubernetes di Amazon EKS e viene fornito come progetto open source. Per ulteriori informazioni, consulta Amazon EKS Distro. Puoi visualizzare e scaricare il codice sorgente per Amazon EKS Distro su GitHub.

Quando scegli le opzioni di implementazione da utilizzare per il cluster Kubernetes, considera quanto segue:

Funzionalità	Amazon EKS	Amazon EKS su Outposts	Amazon EKS Anywhere	Amazon EKS Distro
Hardware	Fornito da AWS	Fornito da AWS	Fornito dall'utente	Fornito dall'utente
Posizione di implementazione	Cloud AWS	Il tuo data center	Il tuo data center	Il tuo data center
Posizione del piano di controllo Kubernetes	Cloud AWS	Cloud AWS o il tuo data center	Il tuo data center	Il tuo data center
Posizione del piano dati Kubernetes	Cloud AWS	Il tuo data center	Il tuo data center	Il tuo data center
Supporto	AWS Support	AWS Support	AWS Support	Supporto della community OSS

Configurazione per l'utilizzo di Amazon EKS

Le risorse AWS in genere prevedono restrizioni di accesso che limitano l'accesso all'entità AWS che le ha create. Pertanto, è fondamentale stabilire sin dall'inizio una corretta configurazione utente nella AWS Command Line Interface. Inoltre, è necessario dotare la macchina locale di strumenti essenziali per una gestione efficiente del cluster Amazon EKS tramite riga di comando. Questo argomento ti aiuterà a prepararti per la gestione del cluster tramite riga di comando.

Fase 1: configurazione della AWS CLI

La AWS CLI è uno strumento a riga di comando per usare i servizi AWS, tra cui Amazon EKS. Viene anche utilizzata per autenticare utenti o ruoli IAM per l'accesso al cluster Amazon EKS e ad altre risorse AWS dalla macchina locale. Per effettuare il provisioning di risorse in AWS dalla riga di comando, è necessario ottenere un ID della chiave di accesso AWS e una chiave segreta da utilizzare nella riga di comando. Quindi è necessario configurare queste credenziali nella AWS CLI. Se non hai già installato la AWS CLI, consulta la sezione Install or update the latest version of the AWS CLI nella Guida per l'utente della AWS Command Line Interface.

Per creare una chiave di accesso

1. Accedi alla AWS Management Console.
2. In alto a destra, scegli il tuo nome utente AWS per aprire il menu di navigazione. Ad esempio, scegli **`webadmin`**. Scegli quindi Credenziali di sicurezza.
3. In Chiavi di accesso, seleziona Crea chiave di accesso.
4. Scegli Interfaccia a riga di comando (CLI), quindi scegli Avanti.
5. Selezionare Create access key (Crea chiave di accesso).
6. Scegli Scarica file .csv.

Per configurare la CLI di AWS CLI

Dopo aver installato la AWS CLI, segui le fasi di seguito per installarla e configurarla. Per ulteriori informazioni, consulta Configura la AWS CLI nella Guida per l'utente di AWS Command Line Interface.

1. In una finestra del terminale, digita il comando riportato qui sotto:

```
aws configure
```

Facoltativamente, puoi configurare un profilo con un nome, ad esempio **`--profile cluster-admin`**. Se configuri un profilo con un nome nella AWS CLI, è necessario passare sempre questo flag nei comandi successivi.

2. Inserisci le credenziali AWS. Per esempio:

```
AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default region name [None]: region-code
Default output format [None]: json
```

Per ottenere un token di sicurezza

Se necessario, esegui il seguente comando per ottenere un nuovo token di sicurezza per la AWS CLI. Per ulteriori informazioni, consulta la sezione `get-session-token` nella Documentazione di riferimento della AWS CLI.

Per impostazione predefinita, il token è valido per 15 minuti. Per modificare il timeout di sessione predefinito, passa il flag **`--duration-seconds`**. Per esempio:

```
aws sts get-session-token --duration-seconds 3600
```

Questo comando restituisce le credenziali di sicurezza temporanee per una sessione della AWS CLI. Dovresti vedere il seguente messaggio di risposta:

```
{
    "Credentials": {
        "AccessKeyId": "ASIA5FTRU3LOEXAMPLE",
        "SecretAccessKey": "JnKgvwfqUD9mNsPoi9IbxAYEXAMPLE",
        "SessionToken": "VERYLONGSESSIONTOKENSTRING",
        "Expiration": "2023-02-17T03:14:24+00:00"
    }
}
```

Per verificare l'identità dell'utente

Se necessario, esegui il comando seguente per verificare le credenziali AWS dell'identità utente IAM (ad esempio *ClusterAdmin*) per la sessione del terminale.

```
aws sts get-caller-identity
```

Questo comando restituisce il nome della risorsa Amazon (ARN) dell'entità IAM configurata per la AWS CLI. Dovresti vedere il seguente esempio di risposta:

```
{
    "UserId": "AKIAIOSFODNN7EXAMPLE",
    "Account": "01234567890",
    "Arn": "arn:aws:iam::01234567890:user/ClusterAdmin"
}
```

Fase 2: installare gli strumenti Kubernetes

Per comunicare con un cluster Kubernetes avrai bisogno di uno strumento per interagire con l'API Kubernetes. Inoltre, sono necessari alcuni strumenti aggiuntivi, ad esempio uno per gestire gli ambienti Kubernetes sulla macchina locale.

Per creare le risorse AWS

- Risorse del cluster Amazon EKS: se non hai esperienza con AWS, ti consigliamo di installare `eksctl`. `eksctl` è un'utilità Infrastructure as Code (IaC) che utilizza AWS CloudFormation per creare facilmente un cluster Amazon EKS. Crea anche risorse Kubernetes aggiuntive, come account di servizio. Per istruzioni sull'installazione di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.
- Risorse AWS: se sei abituato ad automatizzare il provisioning e l'implementazione della tua infrastruttura AWS, ti consigliamo di installare Terraform. Terraform è uno strumento open source infrastructure as code (IaC) sviluppato da HashiCorp. Consente di definire e fornire l'infrastruttura utilizzando un linguaggio di configurazione di alto livello come HashiCorp Configuration Language (HCL) o JSON. Per istruzioni sull'installazione di Terraform, consulta la sezione Install Terraform nella documentazione di Terraform.

Per installare **kubectl**

`kubectl` è uno strumento a riga di comando open source utilizzato per comunicare con il server API di Kubernetes sul cluster Amazon EKS. Se non è già installato sulla macchina locale, scegli una delle seguenti opzioni.

- Versioni AWS: per installare una versione di `kubectl` supportata da Amazon EKS, consulta la sezione Installazione o aggiornamento di `kubectl`.
- Versioni della community: per installare la versione della community più recente di `kubectl`, consulta la pagina Install tools nella documentazione di Kubernetes.

Per configurare un ambiente di sviluppo

- Strumento di implementazione locale: se non hai esperienza con Kubernetes, valuta la possibilità di installare uno strumento di implementazione locale come `minikube` o `kind`. Questi strumenti ti consentono di gestire un cluster Amazon EKS sulla macchina locale.
- Package manager: Helm è un celebre gestore di pacchetti per Kubernetes che semplifica l'installazione e la gestione di pacchetti complessi. Grazie a Helm è più facile installare e gestire pacchetti, come l'AWS Load Balancer Controller, sul cluster Amazon EKS.

Passaggi successivi

- Guida introduttiva ad Amazon EKS

Installazione o aggiornamento di **kubectl**

`Kubectl` è uno strumento a riga di comando utilizzato per comunicare con il server di API Kubernetes. Il file binario `kubectl` è disponibile in molti gestori di pacchetti del sistema operativo. L'utilizzo di un gestore di pacchetti per l'installazione è spesso più semplice rispetto al download e al processo di installazione manuale.

In questa sezione viene descritto come scaricare e installare o aggiornare il file binario `kubectl` sul tuo dispositivo. Il file binario è identico alle versioni della community upstream. Il file binario non è esclusivo di Amazon EKS o AWS.

> **Note**
>
> Utilizza la versione secondaria `kubectl` immediatamente precedente a quella del piano di controllo del cluster Amazon EKS. Ad esempio, un client `kubectl 1.29` funziona con i cluster Kubernetes `1.28`, `1.29` e `1.30`.

Installazione o aggiornamento di **`kubectl`**.

1. Determina se `kubectl` è già installato sul tuo dispositivo.

   ```
   kubectl version --client
   ```

   Se `kubectl` è già installato nel percorso del tuo dispositivo, l'output di esempio include informazioni simili alle seguenti. Se desideri aggiornare la versione correntemente installata con una versione più recente, completa la fase successiva assicurandoti di installare la nuova versione nella stessa posizione in cui si trova la versione corrente.

   ```
   Client Version: v1.30.X-eks-1234567
   ```

   Se non ricevi alcun output, allora `kubectl` non è installato o è installato in una posizione che non si trova nel percorso del dispositivo.
2. Installazione o aggiornamento di `kubectl` sui sistemi operativi `macOS`, Linux e Windows.

   macOS

   Installazione o aggiornamento di **`kubectl`** su **`macOS`**

   1. Scarica il file binario per la versione Kubernetes del cluster da Amazon S3.
      - Kubernetes `1.30`

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/darwin/amd64/kubectl
        ```

      - Kubernetes `1.29`

        ```
        curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/darwin/amd64/kubectl
        ```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/darwin/amd64/kubectl
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/darwin/amd64/kubectl
```

2. (Facoltativo) Verifica il file binario scaricato con la rispettiva somma di controllo `SHA-256`.

a. Scarica la somma di controllo SHA-256 per la versione Kubernetes del tuo cluster.

- Kubernetes 1.30

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.29

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.28

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.27

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.26

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.25

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.24

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes 1.23

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/darwin/amd64/kubectl.sha256
```

b. Controlla la somma di controllo `SHA-256` del file binario scaricato.

```
openssl sha1 -sha256 kubectl
```

c. Assicurati che la somma di controllo generata nell'output corrisponda alla la somma di controllo nel file scaricato `kubectl.sha256`.

3. Applica le autorizzazioni di esecuzione al file binario.

```
chmod +x ./kubectl
```

4. Copia il file binario in una cartella nel tuo `PATH`. Se disponi già di una versione installata di `kubectl`, consigliamo di creare un `$HOME/bin/kubectl` e verificare che `$HOME/bin` venga per primo in `$PATH`.

```
mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:$PATH
```

5. (Facoltativo) Aggiungi il percorso `$HOME/bin` al file di inizializzazione dello shell (interprete di comandi), in modo che sia configurato all'apertura di quest'ultimo.

```
echo 'export PATH=$HOME/bin:$PATH' >> ~/.bash_profile
```

Linux (amd64)

Installazione o aggiornamento di **`kubectl`** su Linux (**`amd64`**)

1. Scarica il file binario `kubectl` per la versione Kubernetes del cluster da Amazon S3.

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl
```

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/amd64/kubectl
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/amd64/kubectl
```

2. (Facoltativo) Verifica il file binario scaricato con il relativo checksum `SHA-256`.
   a. Scarica la somma di controllo `SHA-256` per la versione Kubernetes del cluster da Amazon S3utilizzando il comando per la tua piattaforma hardware. Il primo link per ogni versione è per `amd64`, mentre il secondo link è per `arm64`.
      - Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl.sha256
```

      - Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/amd64/kubectl.sha256
```

      - Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/amd64/kubectl.sha256
```

      - Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/amd64/kubectl.sha256
```

      - Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/amd64/kubectl.sha256
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/amd64/kubectl.sha256
```

b. Controlla la somma di controllo SHA-256 del file binario scaricato con uno dei seguenti comandi.

- ```
  sha256sum -c kubectl.sha256
  ```

  Quando utilizzi questo comando, assicurati che sia visualizzato il seguente output:
  ```

```
kubectl: OK
```

- ```
  openssl sha1 -sha256 kubectl
  ```

  Quando utilizzi questo comando, assicurati che la somma di controllo generata nell'output corrisponda alla la somma di controllo nel file scaricato `kubectl.sha256`.

3. Applica le autorizzazioni di esecuzione al file binario.

   ```
   chmod +x ./kubectl
   ```

4. Copia il file binario in una cartella nel tuo `PATH`. Se disponi già di una versione installata di `kubectl`, consigliamo di creare un `$HOME/bin/kubectl` e verificare che `$HOME/bin` venga per primo in `$PATH`.

   ```
   mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:
   $PATH
   ```

5. (Facoltativo) Aggiungi il percorso `$HOME/bin` al file di inizializzazione dello shell (interprete di comandi), in modo che sia configurato all'apertura di quest'ultimo.

   > **Note**
   >
   > Questa fase prevede l'utilizzo dello shell (interprete di comandi) Bash; se stai utilizzando un altro shell, modifica il comando per utilizzare il file di inizializzazione dello shell in uso.

   ```
   echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
   ```

Linux (arm64)

Installazione o aggiornamento di **`kubectl`** su Linux (**`arm64`**)

1. Scarica il file binario `kubectl` per la versione Kubernetes del cluster da Amazon S3.
   - Kubernetes `1.30`
   ```

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl
```

- Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl
```

- Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/arm64/kubectl
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/arm64/kubectl
```

2. (Facoltativo) Verifica il file binario scaricato con il relativo checksum `SHA-256`.
   a. Scarica la somma di controllo `SHA-256` per la versione Kubernetes del cluster da Amazon S3utilizzando il comando per la tua piattaforma hardware. Il primo link per ogni versione è per `amd64`, mentre il secondo link è per `arm64`.
      - Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl.sha256
```

      - Kubernetes `1.30`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/linux/arm64/kubectl.sha256
```

      - Kubernetes `1.29`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/linux/arm64/kubectl.sha256
```

      - Kubernetes `1.28`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/linux/arm64/kubectl.sha256
```

      - Kubernetes `1.27`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.26`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.25`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.24`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.23`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.22`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.22.17/2024-04-19/bin/linux/arm64/kubectl.sha256
```

- Kubernetes `1.21`

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.21.14/2024-04-19/bin/linux/arm64/kubectl.sha256
```

b. Controlla la somma di controllo SHA-256 del file binario scaricato con uno dei seguenti comandi.

-
```
sha256sum -c kubectl.sha256
```

  Quando utilizzi questo comando, assicurati che sia visualizzato il seguente output:

```
kubectl: OK
```

- ```
  openssl sha1 -sha256 kubectl
  ```

  Quando utilizzi questo comando, assicurati che la somma di controllo generata nell'output corrisponda alla la somma di controllo nel file scaricato `kubectl.sha256`.

3. Applica le autorizzazioni di esecuzione al file binario.

   ```
   chmod +x ./kubectl
   ```

4. Copia il file binario in una cartella nel tuo `PATH`. Se disponi già di una versione installata di `kubectl`, consigliamo di creare un `$HOME/bin/kubectl` e verificare che `$HOME/bin` venga per primo in `$PATH`.

   ```
   mkdir -p $HOME/bin && cp ./kubectl $HOME/bin/kubectl && export PATH=$HOME/bin:
   $PATH
   ```

5. (Facoltativo) Aggiungi il percorso `$HOME/bin` al file di inizializzazione dello shell (interprete di comandi), in modo che sia configurato all'apertura di quest'ultimo.

   > **Note**
   >
   > Questa fase prevede l'utilizzo dello shell (interprete di comandi) Bash; se stai utilizzando un altro shell, modifica il comando per utilizzare il file di inizializzazione dello shell in uso.

   ```
   echo 'export PATH=$HOME/bin:$PATH' >> ~/.bashrc
   ```

Windows

Installazione o aggiornamento di **`kubectl`** su Windows

1. Apri un terminale PowerShell.
2. Scarica il file binario `kubectl` per la versione Kubernetes del cluster da Amazon S3.
   - Kubernetes `1.30`
   ```

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.30.0/2024-05-12/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.29`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.29.3/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.28`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.28.8/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.27`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.27.12/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.26`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.26.15/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.25`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.25.16/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.24`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.24.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.23`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-eks/1.23.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.22`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.22.17/2024-04-19/bin/windows/amd64/kubectl.exe
```

- Kubernetes `1.21`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.21.14/2024-04-19/bin/windows/amd64/kubectl.exe
```

3. (Facoltativo) Verifica il file binario scaricato con il relativo checksum `SHA-256`.
   a. Scarica la somma di controllo `SHA-256` per la versione Kubernetes di Windows.
      - Kubernetes `1.30`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.30.0/2024-05-12/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes `1.29`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.29.3/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes `1.28`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.28.8/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes `1.27`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.27.12/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes `1.26`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.26.15/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

      - Kubernetes `1.25`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.25.16/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.24`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.24.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.23`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.23.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.22`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.22.17/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

- Kubernetes `1.21`

```
curl.exe -O https://s3.us-west-2.amazonaws.com/amazon-
eks/1.21.14/2024-04-19/bin/windows/amd64/kubectl.exe.sha256
```

b. Controlla il checksum `SHA-256` del file binario scaricato.

```
Get-FileHash kubectl.exe
```

c. Assicurati che il checksum generato nell'output corrisponda al checksum nel file `kubectl.sha256` scaricato. L' PowerShelloutput deve essere una stringa di caratteri equivalente in maiuscolo.

4. Copia il file binario in una cartella nel tuo `PATH`. Se disponi di una directory esistente nel `PATH`dedicata alle utility a riga di comando, copia il file binario in questa directory. In alternativa, completa la procedura seguente.

   a. Crea una nuova directory per i file binari della riga di comando, ad esempio `C:\bin`.

   b. Copia il file binario `kubectl.exe` nella nuova directory.

   c. Modifica la variabile di ambiente `PATH` dell'utente o del sistema per aggiungere la nuova directory a `PATH`.

   d. Chiudi il terminale PowerShell e aprine uno nuovo per rendere effettiva la nuova variabile `PATH`.

3. Dopo l'installazione di `kubectl`, puoi verificarne la versione.

```
kubectl version --client
```

Alla prima installazione, `kubectl` non è ancora configurato per comunicare con alcun server. Tratteremo questa configurazione, se necessario, in altre procedure. Per aggiornare la configurazione per comunicare con un particolare cluster, è possibile eseguire il comando seguente. *`region-code`*Sostituiscilo con quello in Regione AWS cui si trova il tuo cluster. Sostituisci *`my-cluster`* con il nome del cluster.

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

Guida introduttiva ad Amazon EKS

Prima di consultare le guide introduttive, verifica di aver completato la configurazione necessaria per utilizzare Amazon EKS. Per ulteriori informazioni, consulta Configurazione per l'utilizzo di Amazon EKS.

Sono disponibili due guide di istruzioni rapide per la creazione di un nuovo cluster Kubernetes con nodi in Amazon EKS:

- Guida introduttiva ad Amazon EKS: `eksctl`: questa guida alle operazioni di base consente di installare tutte le risorse necessarie per iniziare a usare Amazon EKS attraverso `eksctl`, una semplice utility a riga di comando per la creazione e la gestione di cluster Kubernetes su Amazon EKS. Al termine del tutorial, sarà creato un cluster Amazon EKS in esecuzione a cui implementare applicazioni. Questo è il modo più rapido e semplice per iniziare a utilizzare Amazon EKS.
- Guida introduttiva ad Amazon EKS AWS Management Console e AWS CLI— Questa guida introduttiva ti aiuta a creare tutte le risorse necessarie per iniziare a usare Amazon EKS utilizzando AWS Management Console and AWS CLI. Al termine del tutorial, sarà creato un cluster Amazon EKS in esecuzione a cui implementare applicazioni. Attraverso questa guida è possibile creare manualmente ogni risorsa richiesta per un cluster Amazon EKS. Le procedure offrono una visibilità completa sul modo in cui ogni risorsa viene creata e sul modo in cui interagiscono tra loro.

Offriamo anche i seguenti riferimenti:

- Per una raccolta curata di tutorial pratici, consulta Navigating Amazon EKS on Community.AWS
- Per esempi di codice, consulta Esempi di codice per Amazon EKS utilizzando AWS gli SDK.

Guida introduttiva ad Amazon EKS: **`eksctl`**

Questa guida consente di creare tutte le risorse necessarie per iniziare a usare Amazon Elastic Kubernetes Service (Amazon EKS) con `eksctl`, una semplice utility a riga di comando per la creazione e la gestione di cluster Kubernetes su Amazon EKS. Al termine di questo tutorial, sarà creato un cluster Amazon EKS in esecuzione a cui implementare applicazioni.

Le procedure descritte in questa guida consentono di creare automaticamente diverse risorse che è necessario creare manualmente quando si crea il cluster utilizzando il comando AWS Management Console. Se preferisci creare manualmente la maggior parte delle risorse per capire meglio come

interagiscono tra loro, usa la AWS Management Console per creare il cluster e l'elaborazione. Per ulteriori informazioni, consulta Guida introduttiva ad Amazon EKS AWS Management Console e AWS CLI.

Prerequisiti

Prima di iniziare questo tutorial, è necessario installare e configurare i seguenti strumenti e risorse necessarie per creare e gestire un cluster Amazon EKS.

- **`kubectl`**: uno strumento a riga di comando per lavorare con i cluster Kubernetes. Per ulteriori informazioni, consulta Installazione o aggiornamento di `kubectl`.
- **`eksctl`**: uno strumento a riga di comando per usare cluster EKS che automatizza molte attività individuali. Per ulteriori informazioni, consulta Installation nella documentazione di `eksctl`.
- Autorizzazioni IAM richieste: il responsabile della sicurezza IAM che stai utilizzando deve disporre delle autorizzazioni per lavorare con i ruoli IAM di Amazon EKS, i ruoli collegati ai servizi AWS CloudFormation, un VPC e le risorse correlate. Per ulteriori informazioni, consulta Operazioni, risorse e chiavi di condizione per Amazon Elastic Container Service for Kubernetes e Utilizzo di ruoli collegati ai servizi nella Guida per l'utente di IAM. È necessario che tutti i passaggi di questa guida siano completati dallo stesso utente. Esegui il comando seguente per controllare l'utente corrente:

```
aws sts get-caller-identity
```

Fase 1: Creazione di cluster e nodi Amazon EKS

> **Important**
>
> Per iniziare nel modo più semplice e rapido possibile, in questo argomento sono inclusi i passaggi per creare un cluster e nodi con impostazioni predefinite. Prima di creare un cluster e i nodi da utilizzare in produzione, è consigliabile acquisire familiarità con tutte le impostazioni e implementare il cluster e nodi con le impostazioni che soddisfano i requisiti. Per ulteriori informazioni, consulta Creazione di un cluster Amazon EKS e Nodi Amazon EKS. Alcune impostazioni possono essere abilitate solo quando crei cluster e nodi.

È possibile creare un cluster con uno dei seguenti tipi di nodo. Per ulteriori informazioni sui tipi di nodi, consultare Nodi Amazon EKS. Dopo aver implementato il cluster, puoi aggiungere altri tipi di nodo.

- Fargate - Linux: seleziona questo tipo di nodo se desideri eseguire le applicazioni Linux su AWS Fargate. Fargate è un motore di calcolo serverless che consente di implementare Pods di Kubernetes senza dover gestire le istanze Amazon EC2.
- Nodi gestiti - Linux: seleziona questo tipo di nodo se desideri eseguire applicazioni Amazon Linux su istanze Amazon EC2. Sebbene la procedura non sia descritta in questa guida, al cluster puoi anche aggiungere nodi autogestiti di Windows e nodi Bottlerocket.

Creare il cluster Amazon EKS con il seguente comando. Sostituire *my-cluster* con il proprio valore. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può essere più lungo di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster. Sostituisci *region-code* con qualsiasi Regione AWS supporto supportato da Amazon EKS. Per un elenco di Regioni AWS, consulta gli endpoint e le quote di Amazon EKS nella guida di riferimento AWS generale.

Fargate – Linux

```
eksctl create cluster --name my-cluster --region region-code --fargate
```

Managed nodes – Linux

```
eksctl create cluster --name my-cluster --region region-code
```

La creazione di cluster richiede diversi minuti. Durante la creazione, vengono generate diverse righe di output. L'ultima riga di output è simile alla seguente riga di esempio.

```
[...]
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

`eksctl` ha creato un file `kubectl config` in `~/.kube` o aggiunto la configurazione del nuovo cluster all'interno di un file `config` esistente in `~/.kube` sul computer.

Una volta completata la creazione del cluster, visualizza lo AWS CloudFormation stack denominato `eksctl-my-cluster-cluster` nella AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation per vedere tutte le risorse che sono state create.

Fase 2: Visualizzazione delle risorse Kubernetes

1. Visualizzare i nodi del cluster.

   ```
   kubectl get nodes -o wide
   ```

   Di seguito viene riportato un output di esempio:

   Fargate – Linux

   ```
   NAME                                                STATUS   ROLES    AGE
      VERSION              INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-
   VERSION                   CONTAINER-RUNTIME
   fargate-ip-192-0-2-0.region-code.compute.internal   Ready    <none>
     8m3s    v1.2.3-eks-1234567   192.0.2.0     <none>        Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   fargate-ip-192-0-2-1.region-code.compute.internal   Ready    <none>
     7m30s   v1.2.3-eks-1234567   192-0-2-1     <none>        Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   ```

   Managed nodes – Linux

   ```
   NAME                                        STATUS   ROLES    AGE    VERSION
             INTERNAL-IP   EXTERNAL-IP   OS-IMAGE         KERNEL-VERSION
          CONTAINER-RUNTIME
   ip-192-0-2-0.region-code.compute.internal   Ready    <none>   6m7s
      v1.2.3-eks-1234567   192.0.2.0     192.0.2.2     Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   ip-192-0-2-1.region-code.compute.internal   Ready    <none>   6m4s
      v1.2.3-eks-1234567   192.0.2.1     192.0.2.3     Amazon Linux 2
     1.23.456-789.012.amzn2.x86_64   containerd://1.2.3
   ```

   Per ulteriori informazioni sugli elementi visualizzati nell'output, consultare Visualizzazione delle risorse Kubernetes.

2. Visualizzazione dei carichi di lavoro in esecuzione nel cluster.

```
kubectl get pods -A -o wide
```

Di seguito viene riportato un output di esempio:

Fargate – Linux

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE   IP
          NODE                                                NOMINATED NODE
 READINESS GATES
kube-system   coredns-1234567890-abcde   1/1     Running   0          18m
  192.0.2.0   fargate-ip-192-0-2-0.region-code.compute.internal   <none>
   <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          18m
  192.0.2.1   fargate-ip-192-0-2-1.region-code.compute.internal   <none>
   <none>
```

Managed nodes – Linux

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE     IP
          NODE                                        NOMINATED NODE   READINESS
 GATES
kube-system   aws-node-12345             1/1     Running   0          7m43s
   192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
kube-system   aws-node-67890             1/1     Running   0          7m46s
   192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-abcde   1/1     Running   0          14m
   192.0.2.3   ip-192-0-2-3.region-code.compute.internal   <none>
 <none>
kube-system   coredns-1234567890-12345   1/1     Running   0          14m
   192.0.2.4   ip-192-0-2-4.region-code.compute.internal   <none>
 <none>
kube-system   kube-proxy-12345           1/1     Running   0          7m46s
   192.0.2.0   ip-192-0-2-0.region-code.compute.internal   <none>
 <none>
kube-system   kube-proxy-67890           1/1     Running   0          7m43s
   192.0.2.1   ip-192-0-2-1.region-code.compute.internal   <none>
 <none>
```

Per ulteriori informazioni sugli elementi visualizzati nell'output, consultare Visualizzazione delle risorse Kubernetes.

Fase 3: Eliminazione del cluster e dei nodi

Dopo aver creato il cluster e i nodi per questo tutorial, è consigliabile eseguire la pulizia mediante l'eliminazione del cluster e dei nodi con il seguente comando. Per eseguire altre operazioni con questo cluster prima di eseguire la pulizia, consultare Passaggi successivi.

```
eksctl delete cluster --name my-cluster --region region-code
```

Passaggi successivi

I seguenti argomenti della documentazione consentono di estendere la funzionalità del cluster.

- Implementare un'applicazione di esempio al cluster.
- Il principale IAM in grado di creare il cluster è l'unico principale che può effettuare chiamate al server API Kubernetes tramite `kubectl` o la AWS Management Console. Se si desidera che altri principali IAM abbiano accesso al cluster, è necessario aggiungerli. Per ulteriori informazioni, consulta Concedi l'accesso alle Kubernetes API e Autorizzazioni richieste.
- Prima di implementare un cluster da utilizzare in produzione, ti consigliamo di acquisire familiarità con tutte le impostazioni di cluster e nodi. Alcune impostazioni, come l'abilitazione dell'accesso SSH ai nodi Amazon EC2, devono essere definite al momento della creazione del cluster.
- Per aumentare la sicurezza del cluster, configura il plugin Amazon VPC Container Networking Interface per utilizzare i ruoli IAM per gli account di servizio.

Guida introduttiva ad Amazon EKS AWS Management Console e AWS CLI

Questa guida ti aiuta a creare tutte le risorse necessarie per iniziare a usare Amazon Elastic Kubernetes Service (Amazon EKS) utilizzando e il. AWS Management Console AWS CLI Con questa guida è possibile creare manualmente ogni risorsa. Al termine di questo tutorial, sarà creato un cluster Amazon EKS in esecuzione a cui implementare applicazioni.

Le procedure descritte in questa guida ti consentono di visualizzare in modo completo il modo in cui ogni risorsa viene creata e il modo in cui le risorse interagiscono tra loro. Se si preferisce creare automaticamente la maggior parte delle risorse, utilizzare la CLI `eksctl` per creare il cluster e i nodi. Per ulteriori informazioni, consulta Guida introduttiva ad Amazon EKS: `eksctl`.

Prerequisiti

Prima di iniziare questo tutorial, è necessario installare e configurare i seguenti strumenti e risorse necessarie per creare e gestire un cluster Amazon EKS.

- AWS CLI— Uno strumento da riga di comando per lavorare con AWS i servizi, incluso Amazon EKS. Per informazioni, consultare Installazione, aggiornamento e disinstallazione di AWS CLI nella Guida per l'utente di AWS Command Line Interface . Dopo aver installato AWS CLI, ti consigliamo di configurarlo anche. Per ulteriori informazioni, consultare Configurazione rapida con `aws configure` nella Guida per l'utente AWS Command Line Interface .
- **`kubectl`**: uno strumento a riga di comando per lavorare con i cluster Kubernetes. Per ulteriori informazioni, consulta Installazione o aggiornamento di `kubectl`.
- Autorizzazioni IAM richieste: il responsabile della sicurezza IAM che stai utilizzando deve disporre delle autorizzazioni per lavorare con i ruoli IAM di Amazon EKS, i ruoli collegati ai servizi AWS CloudFormation, un VPC e le risorse correlate. Per ulteriori informazioni, consulta Operazioni, risorse e chiavi di condizione per Amazon Elastic Kubernetes Service e Utilizzo di ruoli collegati ai servizi nella Guida per l'utente di IAM. È necessario che tutti i passaggi di questa guida siano completati dallo stesso utente. Esegui il comando seguente per controllare l'utente corrente:

  ```
  aws sts get-caller-identity
  ```

- Ti consigliamo di completare la procedura descritta in questo argomento in una shell Bash. In alternativa, puoi apportare alcune modifiche alla tua shell per alcuni comandi di script, come i caratteri di continuazione della riga, e per il modo in cui le variabili vengono impostate e utilizzate. Inoltre, le regole di escape e di utilizzo delle virgolette per la shell (interprete di comandi) potrebbero essere diverse. Per ulteriori informazioni, consulta Usare le virgolette con le stringhe nella Guida per l' AWS CLIutente. AWS Command Line Interface

Fase 1: Creazione del cluster Amazon EKS;

> **Important**
>
> Per iniziare nel modo più semplice e rapido possibile, in questo argomento sono inclusi i passaggi per creare un cluster con impostazioni predefinite. Prima di creare un cluster da utilizzare in produzione, è consigliabile acquisire familiarità con tutte le impostazioni e implementare un cluster con le impostazioni che soddisfano i tuoi requisiti. Per ulteriori informazioni, consulta Creazione di un cluster Amazon EKS. Alcune impostazioni possono essere abilitate solo quando crei cluster.

Per creare il cluster

1. Creare un Amazon VPC con sottoreti pubbliche e private che soddisfino i requisiti Amazon EKS. Sostituisci *region-code* con qualsiasi Regione AWS supportata da Amazon EKS. Per un elenco di Regioni AWS, consulta gli endpoint e le quote di Amazon EKS nella guida di riferimento AWS generale.Puoi sostituire *my-eks-vpc-stack* con un nome a tua scelta.

   ```
   aws cloudformation create-stack \
     --region region-code \
     --stack-name my-eks-vpc-stack \
     --template-url https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml
   ```

   > **Tip**
   >
   > Per un elenco di tutte le risorse create dal comando precedente, aprire la console AWS CloudFormation all'indirizzo https://console.aws.amazon.com/cloudformation. Scegli lo stack *my-eks-vpc-stack*, quindi la scheda Resources (Risorse).

2. Crea un ruolo IAM del cluster e associa la policy gestita Amazon EKS IAM richiesta. Kubernetesi cluster gestiti da Amazon EKS effettuano chiamate ad altri AWS servizi per tuo conto per gestire le risorse che utilizzi con il servizio.

   a. Copiare i seguenti contenuti in un file denominato *eks-cluster-role-trust-policy.json*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Creare il ruolo.

```
aws iam create-role \
  --role-name myAmazonEKSClusterRole \
  --assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
```

c. Allegare la policy IAM gestita da Amazon EKS richiesta al ruolo.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name myAmazonEKSClusterRole
```

3. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.

   Assicurati che quello Regione AWS mostrato nella parte superiore destra della console sia Regione AWS quello in cui desideri creare il cluster. In caso contrario, scegli il menu a discesa accanto al Regione AWS nome e scegli Regione AWS quello che desideri utilizzare.
4. Scegli Add cluster (Aggiungi cluster), quindi scegli Create (Crea). Se non viene visualizzata questa opzione, scegli Clusters (Cluster) nel pannello di navigazione a sinistra.
5. Nella pagina Configure cluster (Configura cluster), completare le seguenti operazioni:

   a. Immettere un nome per il cluster, ad esempio **`my-cluster`**. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può essere più lungo di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.

   b. Per Cluster Service Role, scegli *`MyAmazoneKS ClusterRole`*.

c. Lasciare le impostazioni rimanenti sui valori di default e selezionare Next (Successivo).

6. Nella pagina Specify networking (Specifica rete), procedere come segue:

   a. Scegli l'ID del VPC creato in un passaggio precedente nell'elenco a discesa VPC. Dovrebbe essere simile a *vpc-00x0000x000x0x000* | *my-eks-vpc-stack-VPC*.

   b. Lasciare le impostazioni rimanenti sui valori di default e selezionare Next (Successivo).

7. Nella pagina Configura osservabilità, scegli Avanti.

8. Nella pagina Seleziona componenti aggiuntivi, scegli Successivo.

   Per ulteriori informazioni sui componenti aggiuntivi, consulta la pagina Componenti aggiuntivi Amazon EKS.

9. Nella pagina Configura le impostazioni dei componenti aggiuntivi selezionati, scegli Successivo.

10. Nella pagina Review and create (Rivedi e crea), scegliere Create (Crea).

    A destra del nome del cluster, lo stato del cluster è Creazione in corso per alcuni minuti fino al termine del processo di provisioning del cluster. Non proseguire con il passaggio successivo finché lo stato non è Attivo.

> **Note**
>
> Potresti ricevere un errore che indica che una delle zone di disponibilità nella richiesta non dispone di capacità sufficiente per creare un cluster Amazon EKS. In questo caso, l'output di errore contiene le zone di disponibilità in grado di supportare un nuovo cluster. Riprova a creare il cluster con almeno due sottoreti che si trovano nelle zone di disponibilità supportate per il tuo account. Per ulteriori informazioni, consulta Capacità insufficiente.

Fase 2: Configurazione del computer per comunicare con il cluster

In questa sezione, imparerai a creare un file `kubeconfig` per il cluster. Le impostazioni in questo file abilitano la CLI `kubectl` per comunicare con il cluster.

Configurazione del computer per comunicazione con il cluster

1. Crea o aggiorna un file `kubeconfig` per il cluster. Sostituisci *region-code* con la Regione AWS in cui hai creato il cluster. Sostituisci *my-cluster* con il nome del cluster.

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

Per impostazione predefinita, il file `config` viene creato in `~/.kube` o la configurazione del nuovo cluster viene aggiunta a un file `config` esistente in `~/.kube`.

2. Prova la configurazione.

```
kubectl get svc
```

> **Note**
>
> Se ricevi qualsiasi altro errore di tipo di risorsa o autorizzazione, consulta la sezione Accesso negato o non autorizzato (`kubectl`) nell'argomento relativo alla risoluzione dei problemi.

Di seguito viene riportato un output di esempio:

```
NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
```

Fase 3: Creazione di nodi

> **Important**
>
> Per iniziare nel modo più semplice e rapido possibile, in questo argomento sono inclusi i passaggi per creare nodi con impostazioni predefinite. Prima di creare i nodi da utilizzare in produzione, è consigliabile acquisire familiarità con tutte le impostazioni e implementare i nodi con le impostazioni che soddisfano i tuoi requisiti. Per ulteriori informazioni, consulta Nodi Amazon EKS. Alcune impostazioni possono essere abilitate solo quando crei i nodi.

È possibile creare un cluster con uno dei seguenti tipi di nodo. Per ulteriori informazioni sui tipi di nodi, consultare Nodi Amazon EKS. Dopo aver implementato il cluster, puoi aggiungere altri tipi di nodo.

- Fargate - Linux: scegli questo tipo di nodo se desideri eseguire le applicazioni Linux su AWS Fargate. Fargate è un motore di calcolo serverless che consente di implementare Pods di Kubernetes senza dover gestire le istanze Amazon EC2.
- Nodi gestiti - Linux: scegli questo tipo di nodo se desideri eseguire applicazioni Amazon Linux su istanze Amazon EC2. Sebbene la procedura non sia descritta in questa guida, al cluster puoi anche aggiungere nodi autogestiti di Windows e nodi Bottlerocket.

Fargate – Linux

Creazione di un profilo Fargate. Quando i Pods Kubernetes vengono implementati con criteri che corrispondono ai criteri definiti nel profilo, i Podsvengono distribuiti in Fargate.

Creazione di un profilo Fargate

1. Creare un ruolo IAM e allegarlo alla policy gestita IAM di Amazon EKS richiesta. Quando il cluster crea Pods sull'infrastruttura Fargate, i componenti in esecuzione sull'infrastruttura Fargate devono effettuare chiamate alle AWS API per conto dell'utente. In questo modo possono eseguire azioni come estrarre le immagini dei container da Amazon ECR o indirizzare i log ad altri AWS servizi. Il ruolo di esecuzione del Pod Amazon EKS fornisce le autorizzazioni IAM per eseguire questa operazione.

   a. Copia i contenuti seguenti in un file denominato *pod-execution-role-trust-policy*.json. *region-code*Sostituiscilo con Regione AWS quello in cui si trova il cluster. Se desideri utilizzare lo stesso ruolo Regioni AWS in tutti gli ambienti del tuo account, sostituiscilo *region-code* con*. Sostituisci *111122223333* con il tuo ID account e *my-cluster* con il nome del cluster. Se vuoi usare lo stesso ruolo per tutti i cluster dell'account, sostituisci *my-cluster* con *.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
```

```
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Crea un ruolo IAM di esecuzione del Pod.

```
aws iam create-role \
  --role-name AmazonEKSFargatePodExecutionRole \
  --assume-role-policy-document file://"pod-execution-role-trust-policy.json"
```

c. Allegare la policy IAM gestita da Amazon EKS richiesta al ruolo.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSFargatePodExecutionRolePolicy \
  --role-name AmazonEKSFargatePodExecutionRole
```

2. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
3. Nella pagina Clusters (Cluster), scegli il cluster *my-cluster*.
4. Sulla pagina ***my-cluster***, completare le seguenti operazioni:

   a. Scegli la scheda Calcolo.

   b. In Fargate Profiles (Profili Fargate), scegli Add Fargate Profile (Aggiungi profilo Fargate).

5. Nella pagina Configure Fargate Profile (Configura profilo Fargate), esegui le operazioni seguenti:

   a. In Nome, inserisci un nome univoco per il profilo Fargate, ad esempio ***my-profile***.

   b. Per il ruolo di esecuzione del Pod, scegli gli FargatePodExecutionRoleAmazonEks che hai creato in un passaggio precedente.

   c. Seleziona l'elenco a discesa Subnets (Sottoreti) e deseleziona qualsiasi sottorete il cui nome contenga `Public`. Per i Pods in esecuzione su Fargate sono supportate solo le sottoreti private.

   d. Seleziona Successivo.

6. Nella pagina Configure Pod selection (Configura selezione pod), procedi come segue:

   a. Per Namespace (Spazio dei nomi), immettere **`default`**.

   b. Seleziona Successivo.

7. Nella pagina Rivedi e crea, controlla le informazioni relative al profilo Fargate e scegli Crea.

8. Dopo alcuni minuti, lo Stato nella sezione Configurazione del profilo Fargate cambierà da In fase di creazione ad Attivo. Non proseguire con il passaggio successivo finché lo stato non è Attivo.

9. Se prevedi di implementare tutti i Pods in Fargate (nessuno sui nodi Amazon EC2), completa la seguente procedura per creare un altro profilo Fargate ed eseguire il resolver di nomi predefinito (CoreDNS) su Fargate.

   > **Note**
   >
   > Se non si effettua questa operazione, in questa fase non si avrà alcun nodo.

   a. Nella pagina Fargate Profile (Profilo Fargate), scegli *`my-profile`*.

   b. Nella sezione Profili Fargate, scegliere Aggiungi profilo Fargate.

   c. Per Nome, immetti ***`CoreDNS`***.

   d. Per il ruolo di esecuzione del Pod, scegli gli FargatePodExecutionRoleAmazonEks che hai creato in un passaggio precedente.

   e. Seleziona l'elenco a discesa Subnets (Sottoreti) e deseleziona qualsiasi sottorete il cui nome contenga `Public`. I Pods in esecuzione su Fargate supportano solo sottoreti private.

   f. Seleziona Successivo.

   g. Per Namespace (Spazio dei nomi), immettere **`kube-system`**.

   h. Scegliere Match labels (Abbina etichette), quindi Add label (Aggiungi etichetta).

   i. Immettere **`k8s-app`** per Key (Chiave) e **`kube-dns`** per il valore. Ciò è necessario affinché il resolver di nomi predefinito (CoreDNS) venga implementato in Fargate.

   j. Seleziona Successivo.

   k. Nella pagina Rivedi e crea, controlla le informazioni relative al profilo Fargate e scegli Crea.

l. Utilizza il comando seguente per rimuovere l'annotazione `eks.amazonaws.com/compute-type : ec2` di default dai Pods CoreDNS.

```
kubectl patch deployment coredns \
    -n kube-system \
    --type json \
    -p='[{"op": "remove", "path": "/spec/template/metadata/annotations/
eks.amazonaws.com~1compute-type"}]'
```

> **Note**
>
> Il sistema crea e implementa due nodi in base all'etichetta del profilo Fargate aggiunta. In Node groups (Gruppi di nodi) non visualizzerai nulla perché non sono applicabili per nodi Fargate, ma vedrai i nuovi nodi elencati nella scheda Overview (Panoramica).

Managed nodes – Linux

Creare un gruppo di nodi gestito specificando le sottoreti e il ruolo IAM del nodo creato nei passaggi precedenti.

Creazione di un gruppo di nodi gestiti Linux di Amazon EC2

1. Creare un ruolo IAM del nodo e allegarvi la policy gestita IAM di Amazon EKS richiesta. Il `kubelet` daemon del nodo Amazon EKS effettua chiamate alle AWS API per tuo conto. I nodi ricevono le autorizzazioni per queste chiamate API attraverso un profilo dell'istanza IAM e le policy associate.

   a. Copiare i seguenti contenuti in un file denominato *`node-role-trust-policy.json`*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
```

```
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Creare il ruolo IAM del nodo.

```
aws iam create-role \
  --role-name myAmazonEKSNodeRole \
  --assume-role-policy-document file://"node-role-trust-policy.json"
```

c. Allegare al ruolo le policy gestite IAM richieste.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name myAmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name myAmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
  --role-name myAmazonEKSNodeRole
```

2. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
3. Scegliere il nome del cluster creato in Fase 1: Creazione del cluster Amazon EKS;, ad esempio *my-cluster*.
4. Sulla pagina ***my-cluster***, completare le seguenti operazioni:

   a. Scegliere la scheda Compute (Calcolo).

   b. Scegliere Add Node Group (Aggiungi gruppo di nodi).

5. Nella pagina Configure Node Group (Configura gruppo di nodi), effettuare le seguenti operazioni:

   a. Per Nome, immettere un nome univoco per il gruppo di nodi gestiti, ad esempio ***my-nodegroup***. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura.

b. Per il nome del ruolo IAM di Node, scegli il ruolo *MyAmazoneKS* che hai creato in NodeRole un passaggio precedente. Consigliamo che ciascun gruppo di nodi utilizzi il proprio ruolo IAM univoco.

c. Seleziona Successivo.

6. Sulla pagina Set compute and scaling configuration (Impostazione della configurazione di calcolo e dimensionamento) accettare i valori di default e selezionare Next (Successivo).
7. Nella pagina Specify networking (Specifica rete), accettare i valori di default e selezionare Next (Successivo).
8. Nella pagina Rivedi e crea, controlla la configurazione del gruppo di nodi gestiti e scegli Crea.
9. Dopo alcuni minuti, lo Stato nella Configurazione del gruppo di nodi cambierà da In fase di creazione a Attivo. Non proseguire con il passaggio successivo finché lo stato non è Attivo.

Fase 4: Visualizzazione delle risorse

Puoi visualizzare i nodi e i carichi di lavoro Kubernetes.

Visualizzazione di nodi e carichi di lavoro

1. Nel pannello di navigazione a sinistra, seleziona Cluster. Nell'elenco Clusters (Cluster), scegli il nome del cluster che hai creato, ad esempio *my-cluster*.
2. Nella pagina ***my-cluster***, scegli le opzioni seguenti:

   a. Scheda Calcolo: verrà visualizzato l'elenco di nodi che sono stati implementati per il cluster. È possibile scegliere il nome di un nodo per visualizzare ulteriori informazioni su di esso.

   b. Scheda Resources (Risorse): verranno visualizzate tutte le risorse Kubernetes che sono state implementate per impostazione predefinita in un cluster Amazon EKS. Seleziona qualsiasi tipo di risorsa nella console per avere ulteriori informazioni in merito.

Fase 5. Eliminazione delle risorse

Dopo aver creato il cluster e i nodi per questo tutorial, devi eliminare le risorse create. Per eseguire altre operazioni con questo cluster prima di eliminare le risorse, consultare Passaggi successivi.

Eliminazione delle risorse create in questa guida

1. Eliminare qualsiasi gruppo di nodi o i profili Fargate creati.

a. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.

b. Nel pannello di navigazione a sinistra, seleziona Cluster. Nell'elenco di cluster, scegliere *my-cluster*.

c. Scegli la scheda Calcolo.

d. Se hai creato un gruppo di nodi, cegli il gruppo di nodi *my-nodegroup* quindi seleziona Delete (Elimina). Immettere ***my-nodegroup***, quindi scegliere Delete (Elimina).

e. Per ogni profilo Fargate creato, selezionarlo e poi scegliere Delete (Elimina). Immettere il nome del profilo, quindi selezionare Delete (Elimina).

> **Note**
>
> Quando si elimina un secondo profilo Fargate, potrebbe essere necessario attendere il completamento dell'eliminazione del primo.

f. Non continuare fino a quando il gruppo di nodi o i profili Fargate eliminati.

2. Elimina il cluster.

a. Nel pannello di navigazione a sinistra, seleziona Cluster. Nell'elenco di cluster, scegliere *my-cluster*.

b. Scegli Delete cluster (Elimina cluster).

c. Immetti ***my-cluster***, quindi scegli Elimina. Non continuare fino a quando il cluster non sarà stato eliminato.

3. Elimina lo AWS CloudFormation stack VPC che hai creato.

a. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.

b. Scegli lo stack *my-eks-vpc-stack*, quindi seleziona Delete (Elimina).

c. Nella finestra di dialogo di conferma Delete ***my-eks-vpc-stack*** (Elimina my-eks-vpc-stack), scegliere Delete stack (Elimina stack).

4. Elimina i ruoli IAM creati.

a. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.

b. Nel pannello di navigazione a sinistra, seleziona Ruoli.

c. ***Seleziona ogni ruolo che hai creato dall'elenco (myAmazoneksClusterRole, oltre a AmazoneKS o myAmazoneKS). FargatePod ExecutionRole NodeRole*** Scegliere Delete (Elimina), inserire il testo di conferma richiesto, quindi scegliere Delete (Elimina).

Passaggi successivi

I seguenti argomenti della documentazione consentono di estendere la funzionalità del cluster.

- Il principale IAM in grado di creare il cluster è l'unico principale che può effettuare chiamate al server API Kubernetes tramite `kubectl` o la AWS Management Console. Se si desidera che altri principali IAM abbiano accesso al cluster, è necessario aggiungerli. Per ulteriori informazioni, consulta Concedi l'accesso alle Kubernetes API e Autorizzazioni richieste.
- Implementare un'applicazione di esempio al cluster.
- Prima di implementare un cluster da utilizzare in produzione, ti consigliamo di acquisire familiarità con tutte le impostazioni di cluster e nodi. Alcune impostazioni, come l'abilitazione dell'accesso SSH ai nodi Amazon EC2, devono essere definite al momento della creazione del cluster.
- Per aumentare la sicurezza del cluster, configura il plugin Amazon VPC Container Networking Interface per utilizzare i ruoli IAM per gli account di servizio.

Cluster Amazon EKS

Un cluster Amazon EKS; è costituito da due componenti principali:

- Il piano di controllo Amazon EKS;
- I nodi di lavoro Amazon EKS; registrati con il piano di controllo

Il piano di controllo Amazon EKS è composto da nodi che eseguono il software Kubernetes, ad esempio `etcd` e il server API Kubernetes. Il piano di controllo viene eseguito in un account gestito da AWS e l'KubernetesAPI è esposta tramite l'endpoint Amazon EKS associato al cluster. Ogni piano di controllo del cluster Amazon EKS è a tenant singolo e unico, e viene eseguito con il suo set di istanze Amazon EC2.

Tutti i dati archiviati dai `etcd` nodi e dai volumi Amazon EBS associati vengono crittografati utilizzando AWS KMS. Il piano di controllo del cluster viene assegnato in più zone di disponibilità e anticipato da un Network Load Balancer di Elastic Load Balancing. Amazon EKS esegue inoltre il provisioning di interfacce di rete elastiche nelle sottoreti VPC per fornire la connettività dalle istanze del piano di controllo (control-plane) ai nodi (ad esempio, per supportare i flussi di dati `kubectl exec`, `logs`, `proxy`).

> **Important**
>
> Nell'ambiente Amazon EKS, l'archiviazione `etcd` è limitata a 8 GiB come da linee guida per l'upstream. Eseguendo il comando seguente è possibile monitorare una metrica per la dimensione corrente del database. Se il tuo cluster ha una versione di Kubernetes precedente alla `1.28`, sostituisci *`apiserver_storage_size_bytes`* con quanto segue:
>
> - Kubernetes versione `1.27` e `1.26` – **`apiserver_storage_db_total_size_in_bytes`**
> - Kubernetes versione `1.25` e versioni precedenti – **`etcd_db_total_size_in_bytes`**
>
> ```
> kubectl get --raw=/metrics | grep "apiserver_storage_size_bytes"
> ```

I nodi Amazon EKS vengono eseguiti nel tuo AWS account e si connettono al piano di controllo del cluster tramite l'endpoint del server API e un file di certificato creato per il cluster.

> **Note**
>
> - É possibile scoprire come funzionano i diversi componenti di Amazon EKS in Reti Amazon EKS.
> - Per i cluster connessi, vedere Amazon EKS Connector.

Argomenti

Creazione di un cluster Amazon EKS

Questo argomento offre una panoramica delle opzioni disponibili e descrivere gli aspetti da considerare quando crei un cluster Amazon EKS. Se devi creare un cluster su un AWS Outpost, vediCluster locali per Amazon EKS su AWS Outposts. Se è la prima volta che crei un cluster Amazon EKS, consigliamo di seguire le nostre guide Guida introduttiva ad Amazon EKS. Le guide consentono di creare un cluster predefinito in modo semplice, senza approfondire tutte le opzioni disponibili.

Prerequisiti

- Un VPC esistente e sottoreti che soddisfano i requisiti di Amazon EKS. Prima di implementare un cluster da utilizzare in produzione, ti consigliamo di approfondire le nozioni relative ai requisiti del VPC e delle sottoreti. Se non disponi di un VPC e di sottoreti, puoi crearli utilizzando un modello fornito da Amazon EKS. AWS CloudFormation

- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul tuo dispositivo o. AWS CloudShell Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella tua home directory nella Guida AWS CLI per l'AWS CloudShell utente.
- Un principale IAM con autorizzazioni per `create` e `describe` un cluster Amazon EKS. Per ulteriori informazioni, consulta Creazione di un cluster Kubernetes locale su un Outpost e Elencare o descrivere tutti i cluster.

Creazione di un cluster Amazon EKS

1. Se disponi già di un ruolo IAM del cluster o intendi creare il cluster con `eksctl`, ignora questo passaggio. Per impostazione predefinita, `eksctl` crea un ruolo per te.

   Per creare un ruolo IAM del cluster Amazon EKS

   1. Per creare un file JSON della policy di attendibilità IAM, esegui il comando seguente.

```
cat >eks-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
```

```
    }
  ]
}
EOF
```

2. Crea il ruolo IAM del cluster Amazon EKS. Se necessario, inserisci il prefisso *eks-cluster-role-trust-policy.json* nel percorso del computer in cui hai scritto il file nella fase precedente. Il comando associa il criterio di attendibilità creato nella fase precedente al ruolo. Per creare un ruolo IAM, l'azione `iam:CreateRole` (autorizzazione) deve essere assegnata al principale IAM che sta creando il ruolo.

```
aws iam create-role --role-name myAmazonEKSClusterRole --assume-role-policy-
document file://"eks-cluster-role-trust-policy.json"
```

3. Puoi assegnare la policy gestita da Amazon EKS o creare una policy personalizzata. Per conoscere le autorizzazioni minime che devi utilizzare nella policy personalizzata, consulta Ruolo IAM del cluster Amazon EKS.

   Allega al ruolo la policy gestita da Amazon EKS, denominata `AmazonEKSClusterPolicy`. Per allegare una policy IAM a un principale IAM, è necessario assegnare al principale che sta allegando la policy una delle azioni IAM (autorizzazioni) seguenti: `iam:AttachUserPolicy` o `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name myAmazonEKSClusterRole
```

2. Crea un cluster Amazon EKS.

   È possibile creare un cluster utilizzando`eksctl`, il AWS Management Console, o il AWS CLI.

   eksctl

   Prerequisito

   La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

   Creazione di un cluster

Crea un cluster `IPv4` di Amazon EKS con la versione più recente di Kubernetes di Amazon EKS supportata nella Regione AWS predefinita. Prima di eseguire il comando, apporta le sostituzioni seguenti:

- Sostituiscilo *`region-code`* con Regione AWS quello in cui desideri creare il cluster.
- Sostituisci *`my-cluster`* con un nome da assegnare al cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può contenere più di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
- Sostituisci *`1.29`* con qualsiasi versione supportata da Amazon EKS.

  > **Note**
  >
  > Per implementare un cluster 1.30 in questo momento, è necessario utilizzare il AWS Management Console o il. AWS CLI

- Modifica i valori di `vpc-private-subnets` in base alle tue esigenze. Puoi inoltre aggiungere altri ID. Devi specificare almeno due ID sottorete, tuttavia, se preferisci specificare le sottoreti pubbliche, puoi modificare `--vpc-private-subnets` in `--vpc-public-subnets`. Alle sottoreti pubbliche è associata una tabella di instradamento con un instradamento a un gateway Internet, al contrario delle sottoreti private. Per questo motivo, ti consigliamo di utilizzare le sottoreti private quando possibile.

  Le sottoreti scelte devono soddisfare i requisiti delle sottoreti di Amazon EKS. Prima di selezionare le sottoreti, tconsigliamo di acquisire familiarità con tutti i requisiti e le considerazioni su cluster VPC e sottoreti di Amazon EKS.

```
eksctl create cluster --name my-cluster --region region-code --version 1.29 --
vpc-private-subnets subnet-ExampleID1,subnet-ExampleID2 --without-nodegroup
```

Il provisioning del cluster richiede diversi minuti. Durante la creazione del cluster, vengono visualizzate diverse righe di output. L'ultima riga di output è simile alla seguente riga di esempio.

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

> **Tip**
>
> Per visualizzare la maggior parte delle opzioni che è possibile specificare durante la creazione di un cluster con `eksctl`, utilizza il comando **`eksctl create cluster --help`**. Per visualizzare tutte le opzioni disponibili, puoi utilizzare un file `config`. Per ulteriori informazioni, consulta Uso dei file config e lo Schema dei file config nella documentazione di `eksctl`. Gli esempi di file di configurazione sono disponibili su GitHub.

Impostazioni facoltative

Di seguito sono riportate le impostazioni facoltative da aggiungere al comando precedente, quando necessario. Puoi abilitare queste opzioni esclusivamente durante la creazione del cluster, non in seguito. Per specificare queste opzioni, non usare il comando precedente. Crea invece il cluster con un file di configurazione `eksctl` e specifica tali impostazioni.

- Se vuoi specificare uno o più gruppi di sicurezza assegnati da Amazon EKS alle interfacce di rete create, scegli l'opzione `securityGroup`.

  Indipendentemente dal fatto che tu scelga o meno un gruppo di sicurezza, Amazon EKS ne crea uno nuovo per consentire la comunicazione tra il cluster e il VPC. Amazon EKS associa questo gruppo di sicurezza, e altri gruppi eventualmente scelti dall'utente, alle interfacce di rete create. Per ulteriori informazioni sul gruppo di sicurezza del cluster creato da Amazon EKS, consulta the section called “Requisiti relativi al gruppo di sicurezza”. Puoi modificare le regole nel gruppo di sicurezza del cluster creato da Amazon EKS.
- Se desideri specificare il blocco di routing interdominio senza classi (CIDR) `IPv4` da cui Kubernetes assegna gli indirizzi IP del servizio, scegli l'opzione `serviceIPv4CIDR`.

  L'indicazione di un intervallo personalizzato consente di evitare conflitti tra servizi Kubernetes e altre reti con peering o connesse al VPC. Immetti un intervallo in una notazione CIDR. Ad esempio: `10.2.0.0/16`.

  Il blocco CIDR deve soddisfare i seguenti requisiti:

  - Essere compreso in uno degli intervalli seguenti: `10.0.0.0/8` `172.16.0.0/12` o `192.168.0.0/16`.
  - Avere una dimensione minima di `/24` e una dimensione massima di `/12`.

- Non sovrapporsi all'intervallo del VPC per le risorse Amazon EKS.

  Puoi specificare questa opzione solo con la famiglia di indirizzi `IPv4` e solo durante la creazione del cluster. Se non specifichi un'opzione, Kubernetes assegna gli indirizzi IP del servizio dai blocchi CIDR `10.100.0.0/16` o `172.20.0.0/16`.

- Se stai creando un cluster e vuoi che il cluster assegni gli indirizzi `IPv6` a Pods e a servizi anziché a indirizzi `IPv4`, scegli l'opzione `ipFamily`.

  Per impostazione predefinita, Kubernetes assegna indirizzi `IPv4` a Pods e servizi. Prima di utilizzare la famiglia `IPv6`, assicurati di conoscere tutte le considerazioni e i requisiti indicati negli argomenti the section called “Considerazioni e requisiti relativi al VPC”, the section called “Considerazioni e requisiti relativi alle sottoreti”, the section called “Requisiti relativi al gruppo di sicurezza” e the section called “IPv6”. Se scegli la famiglia `IPv6`, non potrai specificare un intervallo di indirizzi da cui Kubernetes assegna gli indirizzi del servizio `IPv6`, come avviene invece con la famiglia `IPv4`. Kubernetes assegna gli indirizzi del servizio dall'intervallo di indirizzi locali univoco (`fc00::/7`).

AWS Management Console

Creazione di un cluster

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Seleziona Aggiungi cluster e quindi Crea.
3. Nella pagina Configura cluster compila i campi seguenti:
   - Nome: un nome per il cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra maiuscole e minuscole), trattini e caratteri di sottolineatura. Deve iniziare con un carattere alfanumerico e non può contenere più di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
   - Versione di Kubernetes: la versione di Kubernetes da utilizzare per il cluster. È preferibile selezionare la versione più recente, a meno che non occorra una versione precedente.
   - Ruolo del servizio cluster: scegli il ruolo IAM del cluster Amazon EKS che hai creato per consentire al piano di Kubernetes controllo di gestire AWS le risorse per tuo conto.

- Crittografia dei segreti (facoltativo): scegli di abilitare la crittografia dei segreti Kubernetes tramite una chiave KMS. Puoi abilitare questa funzionalità anche dopo la creazione del cluster. Assicurati, tuttavia, di acquisire familiarità con le informazioni contenute nella sezione Abilitazione della crittografia segreta dei dati in transito su un cluster esistente.
- Tag (Facoltativo): aggiunge eventuali tag al cluster. Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Amazon EKS.

  Al termine, seleziona Avanti.

4. Nella pagina Specifica reti, seleziona i valori dei campi riportati di seguito:
   - VPC: crea il cluster in un VPC esistente conforme ai requisiti su VPC di Amazon EKS. Prima di scegliere un VPC, ti consigliamo di acquisire familiarità con i requisiti e le considerazioni nella sezione Requisiti e considerazioni su VPC e sottoreti di Amazon EKS. Dopo la creazione del cluster, non potrai più modificare il VPC da utilizzare. Se l'elenco risulta vuoto, è necessario crearne uno prima. Per ulteriori informazioni, consulta Creazione di un VPC per il cluster Amazon EKS.
   - Sottoreti: per impostazione predefinita, tutte le sottoreti disponibili nel VPC specificato nel campo precedente sono preselezionate. È necessario selezionarne almeno due.

     Le sottoreti scelte devono soddisfare i requisiti delle sottoreti di Amazon EKS. Prima di selezionare le sottoreti, tconsigliamo di acquisire familiarità con tutti i requisiti e le considerazioni su cluster VPC e sottoreti di Amazon EKS.

     Gruppi di sicurezza: (facoltativo) specifica uno o più gruppi di sicurezza da associare alle interfacce di rete create da Amazon EKS.

     Indipendentemente dal fatto che tu scelga o meno un gruppo di sicurezza, Amazon EKS ne crea uno nuovo per consentire la comunicazione tra il cluster e il VPC. Amazon EKS associa questo gruppo di sicurezza, e altri gruppi eventualmente scelti dall'utente, alle interfacce di rete create. Per ulteriori informazioni sul gruppo di sicurezza del cluster creato da Amazon EKS, consulta the section called “Requisiti relativi al gruppo di sicurezza”. Puoi modificare le regole nel gruppo di sicurezza del cluster creato da Amazon EKS.
   - Scelta della famiglia di indirizzi IP del cluster: puoi scegliere tra IPv4 e IPv6.

     Per impostazione predefinita, Kubernetes assegna indirizzi `IPv4` a Pods e servizi. Prima di utilizzare la famiglia `IPv6`, assicurati di conoscere tutte le considerazioni e i

requisiti indicati negli argomenti the section called “Considerazioni e requisiti relativi al VPC”, the section called “Considerazioni e requisiti relativi alle sottoreti”, the section called “Requisiti relativi al gruppo di sicurezza” e the section called “IPv6”. Se scegli la famiglia `IPv6`, non potrai specificare un intervallo di indirizzi da cui Kubernetes assegna gli indirizzi del servizio `IPv6`, come avviene invece con la famiglia `IPv4`. Kubernetes assegna gli indirizzi del servizio dall'intervallo di indirizzi locali univoco (`fc00::/7`).

- (Facoltativo) Scegli Configura l'intervallo di indirizzi IP del servizio Kubernetes e specifica un Intervallo del servizio **`IPv4`**.

  L'indicazione di un intervallo personalizzato consente di evitare conflitti tra servizi Kubernetes e altre reti con peering o connesse al VPC. Immetti un intervallo in una notazione CIDR. Ad esempio: `10.2.0.0/16`.

  Il blocco CIDR deve soddisfare i seguenti requisiti:

  - Essere compreso in uno degli intervalli seguenti: `10.0.0.0/8` `172.16.0.0/12` o `192.168.0.0/16`.
  - Avere una dimensione minima di `/24` e una dimensione massima di `/12`.
  - Non sovrapporsi all'intervallo del VPC per le risorse Amazon EKS.

  Puoi specificare questa opzione solo con la famiglia di indirizzi `IPv4` e solo durante la creazione del cluster. Se non specifichi un'opzione, Kubernetes assegna gli indirizzi IP del servizio dai blocchi CIDR `10.100.0.0/16` o `172.20.0.0/16`.

- Per Cluster endpoint access (Accesso all'endpoint del cluster), seleziona un'opzione. Puoi modificare questa opzione dopo aver creato il cluster. Prima di selezionare un'opzione non predefinita, assicurati di familiarizzare con le opzioni e le relative implicazioni. Per ulteriori informazioni, consulta Controllo accessi all'endpoint del cluster Amazon EKS.

  Quando hai finito con questa pagina, seleziona Avanti.

5. (Facoltativo) Nella pagina Configura osservabilità, scegli le opzioni Parametri e Registrazione del piano di controllo da attivare. Per impostazione predefinita, i tipi di log sono disattivati.

   - Per ulteriori informazioni sull'opzione relativa ai parametri Prometheus, consulta Attivazione dei parametri Prometheus durante la creazione di un cluster.

- Per ulteriori informazioni sulle opzioni Registrazione del piano di controllo, consulta Logging del piano di controllo di Amazon EKS.

Al termine, seleziona Avanti.

6. Nella pagina Select add-ons (Seleziona componenti aggiuntivi), scegli i componenti aggiuntivi da aggiungere al tuo cluster. Puoi scegliere tutti i componenti aggiuntivi di Amazon EKS e i componenti aggiuntivi di Marketplace AWS necessari. Se i componenti aggiuntivi di Marketplace AWS da installare non sono elencati, puoi cercare i componenti aggiuntivi di Marketplace AWS disponibili inserendo il testo nella casella di ricerca. Puoi eseguire una ricerca anche per categoria, fornitore o modello di prezzi e quindi scegliere i componenti aggiuntivi dai risultati della ricerca. Quando hai finito con questa pagina, seleziona Avanti.
7. Nella pagina Configura le impostazioni dei componenti aggiuntivi selezionati, seleziona la versione da installare. Dopo la creazione del cluster, puoi eseguire in qualsiasi momento l'aggiornamento a una versione successiva. Puoi aggiornare la configurazione di ogni componente aggiuntivo dopo la creazione del cluster. Per ulteriori informazioni sulla configurazione dei componenti aggiuntivi, consulta Aggiornamento di un componente aggiuntivo. Quando hai finito con questa pagina, seleziona Avanti.
8. Nella pagina Rivedi e crea, controlla le informazioni che hai inserito o selezionato nelle pagine precedenti. Se devi apportare modifiche, seleziona Edit (Modifica). Al termine della configurazione, seleziona Create (Crea). Durante il provisioning del cluster, nel campo Stato viene visualizzato il messaggio CREAZIONE.

> **Note**
>
> Potresti ricevere un messaggio di errore indicante che una delle zone di disponibilità nella richiesta non dispone di capacità sufficiente per creare un cluster Amazon EKS. In questo caso, l'output di errore contiene le zone di disponibilità in grado di supportare un nuovo cluster. Riprova a creare il cluster con almeno due sottoreti che si trovano nelle zone di disponibilità supportate per il tuo account. Per ulteriori informazioni, consulta Capacità insufficiente.

Il provisioning del cluster richiede diversi minuti.

AWS CLI

Creazione di un cluster

1. Crea un cluster con il comando seguente. Prima di eseguire il comando, apporta le modifiche seguenti:
   - Sostituiscila *region-code* con Regione AWS quella in cui desideri creare il cluster.
   - Sostituisci *my-cluster* con un nome da assegnare al cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra maiuscole e minuscole), trattini e caratteri di sottolineatura. Deve iniziare con un carattere alfanumerico e non può contenere più di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
   - Sostituisci *1.30* con qualsiasi versione supportata da Amazon EKS.
   - Sostituisci *111122223333* con il tuo ID account e *myAmazonEKSClusterRole* con il nome del ruolo IAM del cluster.
   - Sostituisci i valori di `subnetIds` con quelli in tuo possesso. Puoi inoltre aggiungere altri ID. Devi specificare almeno due ID sottorete,

     Le sottoreti scelte devono soddisfare i requisiti delle sottoreti di Amazon EKS. Prima di selezionare le sottoreti, tconsigliamo di acquisire familiarità con tutti i requisiti e le considerazioni su cluster VPC e sottoreti di Amazon EKS.
   - Se non vuoi specificare un ID del gruppo di sicurezza, rimuovi `,securityGroupIds=sg-`*ExampleID1* dal comando. Se vuoi specificare uno o più ID del gruppo di sicurezza, sostituisci i valori di `securityGroupIds` con quelli in tuo possesso. Puoi inoltre aggiungere altri ID.

     Indipendentemente dal fatto che tu scelga o meno un gruppo di sicurezza, Amazon EKS ne crea uno nuovo per consentire la comunicazione tra il cluster e il VPC. Amazon EKS associa questo gruppo di sicurezza, e altri gruppi eventualmente scelti dall'utente, alle interfacce di rete create. Per ulteriori informazioni sul gruppo di sicurezza del cluster creato da Amazon EKS, consulta the section called “Requisiti relativi al gruppo di sicurezza”. Puoi modificare le regole nel gruppo di sicurezza del cluster creato da Amazon EKS.

```
aws eks create-cluster --region region-code --name my-cluster --kubernetes-
version 1.30 \
```

```
  --role-arn arn:aws:iam::111122223333:role/myAmazonEKSClusterRole \
  --resources-vpc-config
subnetIds=subnet-ExampleID1,subnet-ExampleID2,securityGroupIds=sg-ExampleID1
```

> **Note**
>
> Potresti ricevere un messaggio di errore indicante che una delle zone di disponibilità nella richiesta non dispone di capacità sufficiente per creare un cluster Amazon EKS. In questo caso, l'output di errore contiene le zone di disponibilità in grado di supportare un nuovo cluster. Riprova a creare il cluster con almeno due sottoreti che si trovano nelle zone di disponibilità supportate per il tuo account. Per ulteriori informazioni, consulta Capacità insufficiente.

Impostazioni facoltative

Di seguito sono riportate le impostazioni facoltative da aggiungere al comando precedente, quando necessario. Puoi abilitare queste opzioni esclusivamente durante la creazione del cluster, non in seguito.

- Se desideri specificare il blocco di instradamento interdominio senza classi (CIDR) `IPv4` da cui Kubernetes assegna gli indirizzi IP del servizio, aggiungi il **`--kubernetes-network-config serviceIpv4Cidr=`*`CIDR block`*** al comando seguente.

  L'indicazione di un intervallo personalizzato consente di evitare conflitti tra servizi Kubernetes e altre reti con peering o connesse al VPC. Immetti un intervallo in una notazione CIDR. Ad esempio: `10.2.0.0/16`.

  Il blocco CIDR deve soddisfare i seguenti requisiti:

  - Essere compreso in uno degli intervalli seguenti: `10.0.0.0/8` `172.16.0.0/12` o `192.168.0.0/16`.
  - Avere una dimensione minima di `/24` e una dimensione massima di `/12`.
  - Non sovrapporsi all'intervallo del VPC per le risorse Amazon EKS.

  Puoi specificare questa opzione solo con la famiglia di indirizzi `IPv4` e solo durante la creazione del cluster. Se non specifichi un'opzione, Kubernetes assegna gli indirizzi IP del servizio dai blocchi CIDR `10.100.0.0/16` o `172.20.0.0/16`.

- Se stai creando un cluster e vuoi che il cluster assegni gli indirizzi IPv6 a Pods e a servizi anziché a indirizzi IPv4, aggiungi **--kubernetes-network-config ipFamily=ipv6** al seguente comando.

  Per impostazione predefinita, Kubernetes assegna indirizzi IPv4 a Pods e servizi. Prima di utilizzare la famiglia IPv6, assicurati di conoscere tutte le considerazioni e i requisiti indicati negli argomenti the section called “Considerazioni e requisiti relativi al VPC”, the section called “Considerazioni e requisiti relativi alle sottoreti”, the section called “Requisiti relativi al gruppo di sicurezza” e the section called “IPv6”. Se scegli la famiglia IPv6, non potrai specificare un intervallo di indirizzi da cui Kubernetes assegna gli indirizzi del servizio IPv6, come avviene invece con la famiglia IPv4. Kubernetes assegna gli indirizzi del servizio dall'intervallo di indirizzi locali univoco (fc00::/7).

2. Il provisioning del cluster richiede diversi minuti. È possibile eseguire query sullo stato del cluster con il comando seguente.

   ```
   aws eks describe-cluster --region region-code --name my-cluster --query
    "cluster.status"
   ```

   Non passare alla fase successiva finché l'output restituito non è ACTIVE.

3. Se hai creato il cluster utilizzando eksctl, puoi ignorare questo passaggio, poiché è già stato completato da eksctl. Abilita kubectl per consentire la comunicazione con il cluster aggiungendo un nuovo contesto al file config kubectl. Per ulteriori informazioni su come creare e aggiornare il file, consulta Creazione o aggiornamento di un file kubeconfig per un cluster Amazon EKS.

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   Di seguito viene riportato un output di esempio:

   ```
   Added new context arn:aws:eks:region-code:111122223333:cluster/my-cluster to /home/
   username/.kube/config
   ```

4. Conferma la comunicazione con il cluster eseguendo il comando seguente.

   ```
   kubectl get svc
   ```

Di seguito viene riportato un output di esempio:

```
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
```

5. (Consigliato) Per utilizzare alcuni componenti aggiuntivi di Amazon EKS o per consentire a singoli Kubernetes carichi di lavoro di avere autorizzazioni specifiche AWS Identity and Access Management (IAM), crea un provider IAM OpenID Connect (OIDC) per il tuo cluster. È necessario creare un provider IAM OIDC per il cluster una sola volta. Per ulteriori informazioni sui componenti aggiuntivi di Amazon EKS, consulta Componenti aggiuntivi Amazon EKS. Per ulteriori informazioni sull'assegnazione di autorizzazioni IAM specifiche ai carichi di lavoro, consulta Ruoli IAM per gli account di servizio.
6. (Consigliato) Configura il cluster per il Amazon VPC CNI plugin for Kubernetes prima di implementare i nodi Amazon EC2 nel cluster. Per impostazione predefinita, il plugin è stato installato con il cluster. Quando aggiungi i nodi Amazon EC2 al cluster, il plugin viene implementato automaticamente su ogni nodo Amazon EC2 aggiunto. Il plugin richiede di collegare una delle policy IAM seguenti a un ruolo IAM:

   Policy IAM gestita da **`AmazonEKS_CNI_Policy`**

   Se il cluster utilizza la famiglia `IPv4`

   Una policy IAM che crei

   Se il cluster utilizza la famiglia `IPv6`

   Il ruolo IAM a cui si collega la policy può essere il ruolo IAM del nodo o un ruolo dedicato utilizzato solo per il plugin. Consigliamo di allegare la policy a quest'ultimo ruolo. Per ulteriori informazioni sulla creazione del ruolo, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA) o Ruolo IAM del nodo Amazon EKS.
7. Se hai distribuito il cluster utilizzando AWS Management Console, puoi saltare questo passaggio. Per impostazione predefinita, la AWS Management Console implementa i componenti aggiuntivi Amazon VPC CNI plugin for Kubernetes, CoreDNS e `kube-proxy` di Amazon EKS.

   Se implementi il tuo cluster tramite `eksctl` o AWS CLI, vengono implementati i componenti aggiuntivi autogestiti Amazon VPC CNI plugin for Kubernetes, CoreDNS e `kube-proxy`.

Puoi eseguire la migrazione dei componenti aggiuntivi autogestiti Amazon VPC CNI plugin for Kubernetes, CoreDNS e `kube-proxy` implementati con il cluster verso i componenti aggiuntivi di Amazon EKS. Per ulteriori informazioni, consulta Componenti aggiuntivi Amazon EKS.

8. (Facoltativo) Se non l'hai già fatto, puoi abilitare i parametri Prometheus per il cluster. Per ulteriori informazioni, consulta Creare uno scraper nella Guida per l'utente di Amazon Managed Service for Prometheus.
9. Se hai abilitato i parametri Prometheus, devi configurare `aws-auth ConfigMap` per fornire allo scraper le autorizzazioni in-cluster. Per ulteriori informazioni, consulta Configurazione del cluster Amazon EKS nella Guida per l'utente di Amazon Managed Service for Prometheus.
10. Se hai intenzione di implementare nel tuo cluster carichi di lavoro che utilizzano volumi Amazon EBS e hai creato un cluster `1.23` o versione successiva, devi installare il Driver CSI per Amazon EBS nel cluster prima di implementare i carichi di lavoro.

Fasi successive consigliate:

- L'utente o il principale IAM che ha creato il cluster è l'unico principale in grado di accedere al cluster. Concedi autorizzazioni ad altri principali IAM in modo che possano accedere al tuo cluster.
- Se il principale IAM che ha creato il cluster dispone solo delle autorizzazioni IAM minime a cui si fa riferimento nei prerequisiti, potresti voler aggiungere ulteriori autorizzazioni Amazon EKS per quel principale IAM. Per ulteriori informazioni su come concedere le autorizzazioni Amazon EKS ali principali IAM, consulta Identity and Access Management per Amazon EKS.
- Se desideri che il principale IAM che ha creato il cluster o qualsiasi altro principale visualizzi le risorse Kubernetes nella console Amazon EKS, concedi Autorizzazioni richieste alle entità.
- Se desideri che i nodi e i principali IAM accedano al cluster dall'interno della VPC, attiva l'endpoint privato per il cluster. L'endpoint pubblico è abilitato per impostazione predefinita. Puoi disabilitare l'endpoint pubblico dopo aver abilitato l'endpoint privato, se lo desideri. Per ulteriori informazioni, consulta Controllo accessi all'endpoint del cluster Amazon EKS.
- Abilitare la crittografia dei segreti per il cluster.
- Configurare la registrazione per il cluster.
- Aggiungi nodi al cluster.

Approfondimenti sui cluster

Gli approfondimenti sui cluster offerti da Amazon EKS forniscono consigli per aiutare a seguire le best practice di Amazon EKS e Kubernetes. Ogni cluster Amazon EKS è soggetto a controlli automatici e ricorrenti basati su un elenco di approfondimenti curato da Amazon EKS. Questi controlli sugli approfondimenti sono completamente gestiti da Amazon EKS e offrono consigli su come affrontare eventuali esiti.

Utilizzo consigliato di Cluster Insights:

- Prima di aggiornare la Kubernetes versione del cluster, controlla cluster insights nella console EKS.
- Se il tuo cluster ha identificato problemi, esaminali e apporta le correzioni appropriate. I problemi includono collegamenti ad Amazon EKS eKubernetes.
- Dopo aver risolto i problemi, attendi che Cluster Insights si aggiorni. Se tutti i problemi sono stati risolti, aggiorna il cluster.

Attualmente, Amazon EKS restituisce solo approfondimenti riguardanti la disponibilità per l'aggiornamento della versione di Kubernetes.

Gli approfondimenti sugli aggiornamenti identificano possibili problemi che possono influire sugli aggiornamenti del cluster Kubernetes. Questo riduce l'impegno richiesto agli amministratori nella preparazione degli aggiornamenti e aumenta l'affidabilità delle applicazioni sulle versioni più recenti di Kubernetes. I cluster vengono scansionati automaticamente da Amazon EKS in base a un elenco di possibili aggiornamenti di versione di Kubernetes che influiscono sui problemi. Amazon EKS aggiorna frequentemente l'elenco dei controlli sugli approfondimenti in base alle revisioni delle modifiche apportate in ogni versione di Kubernetes.

Gli approfondimenti sugli aggiornamenti offerti da Amazon EKS velocizzano il processo di test e verifica per le nuove versioni. Inoltre, consentono agli amministratori dei cluster e agli sviluppatori di applicazioni di avvalersi delle funzionalità di Kubernetes più recenti, evidenziando eventuali problemi e offrendo consigli per porvi rimedio. Per visualizzare l'elenco dei controlli sugli approfondimenti eseguiti e gli eventuali problemi rilevanti identificati da Amazon EKS, puoi richiamare l'operazione dell'API `ListInsights` di Amazon EKS o consultare la console Amazon EKS.

Cluster Insights si aggiorna periodicamente. Non è possibile aggiornare manualmente Cluster Insights. Se risolvi un problema relativo al cluster, l'aggiornamento di Cluster Insights richiederà del tempo. Per determinare se una correzione ha avuto successo, confronta l'ora in cui è stata implementata la modifica con quella «ultima ora di aggiornamento» di Cluster Insight.

Visualizza le informazioni sul cluster (Console)

Per visualizzare le informazioni dettagliate di un cluster Amazon EKS:

a. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
b. Dall'elenco dei cluster, scegli il nome del cluster Amazon EKS per il quale desideri visualizzare gli approfondimenti.
c. Scegli la scheda Approfondimenti sugli aggiornamenti.
d. Nella pagina Approfondimenti sugli aggiornamenti vedrai i seguenti campi:
   - Nome: il controllo eseguito da Amazon EKS sul cluster.
   - Stato dell'approfondimento: un approfondimento che presenta lo stato "Errore" in genere indica che la versione di Kubernetes interessata è N+1 rispetto alla versione corrente del cluster, mentre lo stato "Avviso" indica che l'approfondimento si applica a una versione futura N+2 o superiore di Kubernetes. Un approfondimento con lo stato "Ammesso" indica che Amazon EKS non ha riscontrato alcun problema associato a questo controllo sull'approfondimento nel cluster. Lo stato dell'approfondimento "Sconosciuto" significa che Amazon EKS non è in grado di determinare se il cluster è interessato da questo controllo sull'approfondimento.
   - Versione: la versione di Kubernetes che l'approfondimento ha verificato per individuare possibili problemi.
   - Ora dell'ultimo aggiornamento (UTC-5:00): l'ora in cui lo stato dell'approfondimento è stato aggiornato l'ultima volta per questo cluster.
   - Ora dell'ultima transizione (UTC-5:00): l'ora in cui lo stato di questo approfondimento è cambiato l'ultima volta.
   - Descrizione: informazioni tratte dal controllo sull'approfondimento, che include l'avviso e le azioni consigliate per la correzione.

Visualizza gli approfondimenti sul cluster (AWS CLI)

Per visualizzare le informazioni dettagliate di un cluster Amazon EKS:

a. Determina in quale cluster desideri verificare la presenza di approfondimenti. Il comando seguente elenca gli approfondimenti per un cluster specifico. Apportare le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato:
   - Sostituisci *region-code* con il codice per la tua Regione AWS.
   - Sostituisci *my-cluster* con il nome del cluster.

```
aws eks list-insights --region region-code --cluster-name my-cluster
```

Di seguito viene riportato un output di esempio:

```
{
    "insights": [
        {
            "category": "UPGRADE_READINESS",
            "name": "Deprecated APIs removed in Kubernetes v1.29",
            "insightStatus": {
                "status": "PASSING",
                "reason": "No deprecated API usage detected within the last 30 days."
            },
            "kubernetesVersion": "1.29",
            "lastTransitionTime": 1698774710.0,
            "lastRefreshTime": 1700157422.0,
            "id": "123e4567-e89b-42d3-a456-579642341238",
            "description": "Checks for usage of deprecated APIs that are scheduled
 for removal in Kubernetes v1.29. Upgrading your cluster before migrating to the
 updated APIs supported by v1.29 could cause application impact."
        }
    ]
}
```

b. Eseguire il comando seguente per visualizzare le informazioni dettagliate relative all'approfondimento. Apportare le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato:

- Sostituisci *region-code* con il codice per la tua Regione AWS.
- Sostituisci *123e4567-e89b-42d3-a456-579642341238* con l'ID dell'approfondimento recuperato dall'elenco degli approfondimenti del cluster.
- Sostituisci *my-cluster* con il nome del cluster.

```
aws eks describe-insight --region region-code --id 123e4567-e89b-42d3-
a456-579642341238 --cluster-name my-cluster
```

Di seguito viene riportato un output di esempio:

```
{
    "insight": {
```

```
        "category": "UPGRADE_READINESS",
        "additionalInfo": {
            "EKS update cluster documentation": "https://docs.aws.amazon.com/eks/
latest/userguide/update-cluster.html",
            "Kubernetes v1.29 deprecation guide": "https://kubernetes.io/docs/
reference/using-api/deprecation-guide/#v1-29"
        },
        "name": "Deprecated APIs removed in Kubernetes v1.29",
        "insightStatus": {
            "status": "PASSING",
            "reason": "No deprecated API usage detected within the last 30 days."
        },
        "kubernetesVersion": "1.29",
        "recommendation": "Update manifests and API clients to use newer Kubernetes
 APIs if applicable before upgrading to Kubernetes v1.29.",
        "lastTransitionTime": 1698774710.0,
        "lastRefreshTime": 1700157422.0,
        "categorySpecificSummary": {
            "deprecationDetails": [
                {
                    "usage": "/apis/flowcontrol.apiserver.k8s.io/v1beta2/
flowschemas",
                    "replacedWith": "/apis/flowcontrol.apiserver.k8s.io/v1beta3/
flowschemas",
                    "stopServingVersion": "1.29",
                    "clientStats": [],
                    "startServingReplacementVersion": "1.26"
                },
                {
                    "usage": "/apis/flowcontrol.apiserver.k8s.io/v1beta2/
prioritylevelconfigurations",
                    "replacedWith": "/apis/flowcontrol.apiserver.k8s.io/v1beta3/
prioritylevelconfigurations",
                    "stopServingVersion": "1.29",
                    "clientStats": [],
                    "startServingReplacementVersion": "1.26"
                }
            ]
        },
        "id": "f6a11fe4-77f7-48c6-8326-9a13f022ecb3",
        "resources": [],
        "description": "Checks for usage of deprecated APIs that are scheduled for
 removal in Kubernetes v1.29. Upgrading your cluster before migrating to the updated
 APIs supported by v1.29 could cause application impact."
```

```
    }
}
```

Aggiornamento della versione di Kubernetes del cluster Amazon EKS

Quando è disponibile una nuova versione di Kubernetes in Amazon EKS, potrai aggiornare il cluster Amazon EKS alla versione più recente.

> **Important**
>
> Una volta aggiornato un cluster, non è possibile effettuare il downgrade a una versione precedente. Prima di eseguire l'aggiornamento a una nuova versione di Kubernetes, è consigliabile esaminare le informazioni contenute in Versioni Kubernetes di Amazon EKS e nella procedura di aggiornamento descritta in questo argomento.

Le nuove versioni di Kubernetes hanno introdotto modifiche significative. Pertanto ti consigliamo di verificare il comportamento delle applicazioni rispetto alla nuova versione di Kubernetes prima di eseguire l'aggiornamento sui cluster di produzione. Puoi eseguire questa operazione mediante la creazione di un flusso di lavoro di integrazione continua per testare il comportamento totale dell'applicazione prima di passare a una nuova versione di Kubernetes.

Il processo di aggiornamento consiste nell'avvio, da parte di Amazon EKS, di nuovi nodi del server API con la versione aggiornata di Kubernetes che sostituiscono quelli esistenti. Amazon EKS esegue i controlli dell'integrità dell'infrastruttura standard e dello stato di preparazione per il traffico di rete su questi nuovi nodi per verificare che funzionino come previsto. Tuttavia, una volta avviato l'aggiornamento del cluster, non è possibile metterlo in pausa o interromperlo. Se uno di questi controlli non va a buon fine, Amazon EKS ripristina l'implementazione dell'infrastruttura e il cluster rimane nella versione precedente di Kubernetes. Le applicazioni in esecuzione non sono interessate e il cluster non viene mai lasciato in uno stato non deterministico o irrecuperabile. Amazon EKS esegue regolarmente il backup di tutti i cluster gestiti e, se necessario, dispone di meccanismi per il recupero dei cluster. Stiamo valutando e migliorando costantemente i nostri processi di gestione dell'infrastruttura di Kubernetes.

Per aggiornare il cluster, Amazon EKS richiede fino a cinque indirizzi IP disponibili dalle sottoreti specificate al momento della creazione del cluster. Amazon EKS crea nuove interfacce di rete

elastiche (interfacce di rete) per il cluster in una delle sottoreti specificate, che può essere diversa rispetto a quella in cui si trovano le interfacce di rete esistenti. Assicurati quindi che le regole del gruppo di sicurezza consentano la comunicazione necessaria con il cluster per una delle sottoreti specificate al momento della creazione del cluster. Se questa sottorete non esiste, se non ha abbastanza indirizzi IP disponibili o se non dispone di regole del gruppo di sicurezza per la comunicazione con il cluster, l'aggiornamento potrebbe non riuscire.

> **Note**
>
> Per garantire che l'endpoint del server API per il tuo cluster sia sempre accessibile, Amazon EKS fornisce un'elevata disponibilitàKubernetespiano di controllo ed esegue aggiornamenti continui delle istanze del server API durante le operazioni di aggiornamento. Per tenere conto della modifica degli indirizzi IP delle istanze del server API che supportano il tuoKubernetesEndpoint del server API, devi assicurarti che i client del tuo server API gestiscano le riconnessioni in modo efficace. Versioni recenti di`kubectl` ilKubernetesclientelibrerieche sono ufficialmente supportati, esegui questo processo di riconnessione in modo trasparente.

Aggiornamento della versione di Kubernetes per il cluster Amazon EKS

Aggiornamento della versione di Kubernetes per il cluster

1. Confronta la versione di Kubernetes del piano di controllo (control-plane) del cluster con la versione di Kubernetes dei nodi.

   - Ottieni la versione di Kubernetes del piano di controllo (control-plane) del cluster.

     ```
     kubectl version
     ```

   - Ottieni la versione di Kubernetes dei tuoi nodi. Questo comando restituisce tutti i nodi Amazon EC2 e Fargate gestiti e autogestiti. Ogni Pod Fargate è presente nell'elenco assieme al rispettivo nodo.

     ```
     kubectl get nodes
     ```

   Prima di aggiornare il piano di controllo (control-plane) a una nuova versione di Kubernetes, assicurati che la versione secondaria di Kubernetes dei nodi gestiti e dei nodi Fargate nel cluster

sia la stessa della versione corrente del piano di controllo (control-plane). Ad esempio, se il piano di controllo è in versione in esecuzione `1.29` e uno dei nodi è in esecuzione`1.28`, è necessario aggiornare i nodi alla versione `1.29` prima di aggiornare il piano di controllo alla versione 1.30. Prima di aggiornare il piano di controllo, si consiglia inoltre di aggiornare i nodi autogestiti alla stessa versione del piano di controllo. Per ulteriori informazioni, consulta Aggiornamento di un gruppo di nodi gestiti e Aggiornamenti del nodo autogestito. Se hai nodi Fargate con una versione secondaria inferiore rispetto alla versione del piano di controllo (control-plane), devi prima eliminare il Pod rappresentato dal nodo. a aggiornare il piano di controllo (control-plane). Dopo averli implementati di nuovo, tutti i Pods rimanenti verranno aggiornati alla nuova versione.

2. Se la versione di Kubernetes su cui è stato originariamente implementato il cluster era Kubernetes `1.25` o successiva, puoi ignorare questo passaggio.

   Per impostazione predefinita, il controller di ammissione della policy di sicurezza Pod è abilitato sui cluster Amazon EKS. Assicurarsi che siano presenti le policy di sicurezza dei Pod appropriate prima di eseguire l'aggiornamento del cluster, al fine di evitare problemi. È possibile verificare la policy di default con il comando **`kubectl get psp eks.privileged`**.

   ```
   kubectl get psp eks.privileged
   ```

   Se ricevi il seguente errore, consulta Policy di sicurezza Pod predefinita di Amazon EKS prima di procedere.

   ```
   Error from server (NotFound): podsecuritypolicies.extensions "eks.privileged" not
    found
   ```

3. Se la versione di Kubernetes su cui è stato originariamente implementato il cluster era Kubernetes `1.18` o successiva, puoi ignorare questo passaggio.

   potrebbe essere necessario rimuovere un termine interrotto dal manifesto CoreDNS.

   a. Controllare se il manifesto CoreDNS ha una riga contenente solo la parola `upstream`.

      ```
      kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' |
       grep upstream
      ```

      Se non viene restituito alcun output, il manifesto non presenta alcun riga ed è possibile passare alla fase successiva. Se come risultato viene restituita la parola `upstream` è necessario rimuoverla.

b. Modificare il file configmap, rimuovendo la riga vicino alla parte superiore del file contenente solo la parola `upstream`. Non apportare ulteriori modifiche al file. Dopo aver rimosso la riga, salvare le modifiche.

```
kubectl edit configmap coredns -n kube-system -o yaml
```

4. Aggiorna il cluster utilizzando`eksctl`, il AWS Management Console, o il AWS CLI.

> **Important**
>
> - Se stai eseguendo l'aggiornamento alla versione `1.23` e utilizzi volumi Amazon EBS nel cluster, allora dovrai installare il driver CSI di Amazon EBS nel cluster prima di aggiornare il cluster alla versione `1.23` per evitare interruzioni del carico di lavoro. Per ulteriori informazioni, consulta Kubernetes1.23 e Driver CSI per Amazon EBS.
> - Kubernetes `1.24` e versioni successive utilizzano `containerd` come runtime del container predefinito. Se stai passando al runtime `containerd` e hai già configurato Fluentd per Container Insights, devi eseguire la migrazione di Fluentd a Fluent Bit prima di aggiornare il cluster. I parser Fluentd sono configurati per analizzare solo i messaggi di log in formato JSON. A differenza di `dockerd`, il runtime del container `containerd` contiene messaggi di log che non sono in formato JSON. Se non esegui la migrazione a Fluent Bit, alcuni dei parser Fluentd's configurati genereranno un'enorme quantità di errori all'interno del container Fluentd. Per ulteriori informazioni sulla migrazione, consulta Configurare Fluent Bit come invio DaemonSet di log a Logs. CloudWatch
> - Poiché Amazon EKS viene eseguito in un piano di controllo ad alta disponibilità, è possibile aggiornare solo una versione secondaria alla volta. Per ulteriori informazioni su questo requisito, consulta Versione di Kubernetes e policy per il supporto Skew della versione. Supponiamo che la versione corrente del cluster sia `1.28` e che desideri aggiornarlo alla versione `1.30`. Devi prima aggiornare la versione `1.28` del cluster alla versione `1.29`, quindi aggiornare la versione `1.29` del cluster alla versione `1.30`.
> - Verifica la discrepanza di versione tra `kube-apiserver` e `kubelet` di Kubernetes sui nodi.
>   - A partire dalla versione `1.28` di Kubernetes, `kubelet` può essere fino a tre versioni minori precedenti rispetto a `kube-apiserver`. Consulta la sezione Kubernetes upstream version skew policy.

- Se `kubelet` sui nodi gestiti e Fargate è nella versione `1.25` o più recente di Kubernetes, è possibile aggiornare il cluster fino a tre versioni successive senza aggiornare la versione di `kubelet`. Ad esempio, se `kubelet` è nella versione `1.25`, è possibile aggiornare la versione del cluster Amazon EKS dalla `1.25` alla `1.26`, `1.27` e `1.28` mentre `kubelet` rimane sulla versione `1.25`.
- Se `kubelet` sui nodi gestiti e Fargate è nella versione `1.24` o precedente di Kubernetes, può essere al massimo di due versioni minori precedenti rispetto a `kube-apiserver`. In altre parole, se `kubelet` è nella versione `1.24` o precedente, è possibile aggiornare il cluster solo fino a due versioni successive. Ad esempio, se `kubelet` è nella versione `1.21`, è possibile aggiornare la versione del cluster Amazon EKS dalla `1.21` alla `1.22` e `1.23`, ma non sarà possibile aggiornare il cluster alla `1.24` mentre `kubelet` rimane sulla `1.21`.

- Come best practice prima di iniziare un aggiornamento, assicurati che il `kubelet` sui tuoi nodi sia alla stessa versione di Kubernetes del tuo piano di controllo.
- Se il cluster è configurato con una versione di Amazon VPC CNI plugin for Kubernetes precedente a `1.8.0`, è preferibile aggiornare il plug-in all'ultima versione prima di aggiornare il cluster. Per aggiornare il plug-in, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.
- Se stai aggiornando il cluster alla versione `1.25` o successiva e hai AWS Load Balancer Controller distribuito nel cluster, aggiorna il controller alla versione `2.4.7` o successiva prima di aggiornare la versione del cluster a `1.25`. Per ulteriori informazioni, consulta le note di rilascio di Kubernetes1,25.

eksctl

Questa procedura richiede `eksctl` versione `0.183.0` o successiva. Puoi verificare la versione con il comando seguente:

```
eksctl version
```

Per istruzioni sull'installazione e sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Aggiorna la versione di Kubernetes del tuo piano di controllo di Amazon EKS. Sostituisci *`my-cluster`* con il nome del cluster. Sostituisci *`1.30`* con il numero di versione supportato

da Amazon EKS a cui desideri aggiornare il cluster. Per l'elenco completo delle versioni supportate, consulta Versioni Kubernetes di Amazon EKS.

```
eksctl upgrade cluster --name my-cluster --version 1.30 --approve
```

Il processo di aggiornamento può richiedere alcuni minuti per il completamento.

AWS Management Console

a. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
b. Scegliere il nome del cluster Amazon EKS da aggiornare, quindi selezionare Aggiorna la versione del cluster.
c. Per Kubernetes version (Versione Kubernetes), seleziona la versione a cui aggiornare il cluster e scegli Update (Aggiorna).
d. In Nome cluster, immettere il nome del cluster e scegliere Conferma.

   Il processo di aggiornamento può richiedere alcuni minuti per il completamento.

AWS CLI

a. Aggiornare il cluster Amazon EKS con il seguente comando AWS CLI . Sostituisci i *example values* con i valori in tuo possesso. Sostituisci *1.30* con il numero di versione supportato da Amazon EKS a cui desideri aggiornare il cluster. Per l'elenco completo delle versioni supportate, consulta Versioni Kubernetes di Amazon EKS.

   ```
   aws eks update-cluster-version --region region-code --name my-cluster --
   kubernetes-version 1.30
   ```

   Di seguito viene riportato un output di esempio:

   ```
   {
       "update": {
           "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
           "status": "InProgress",
           "type": "VersionUpdate",
           "params": [
               {
                   "type": "Version",
   ```

```
                "value": "1.30"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
}
```

b. É possibile monitorare lo stato di aggiornamento del cluster attraverso il seguente comando. Utilizzare il nome del cluster e l'ID aggiornamento restituito dal comando precedente. Quando viene visualizzato lo stato `Successful`, l'aggiornamento è completo. Il processo di aggiornamento può richiedere alcuni minuti per il completamento.

```
aws eks describe-update --region region-code --name my-cluster --update-
id b5f0ba18-9a87-4450-b5a0-825e6e84496f
```

Di seguito viene riportato un output di esempio:

```
{
    "update": {
        "id": "b5f0ba18-9a87-4450-b5a0-825e6e84496f",
        "status": "Successful",
        "type": "VersionUpdate",
        "params": [
            {
                "type": "Version",
                "value": "1.30"
            },
            {
                "type": "PlatformVersion",
                "value": "eks.1"
            }
        ],
[...]
        "errors": []
    }
}
```

5. Una volta completato l'aggiornamento del cluster, aggiorna i nodi alla stessa versione secondaria di Kubernetes del cluster aggiornato. Per ulteriori informazioni, consulta Aggiornamenti del nodo autogestito e Aggiornamento di un gruppo di nodi gestiti. Tutti i nuovi Pods avviati in Fargate hanno una versione di `kubelet` che corrisponde alla versione del cluster. I Pods Fargate esistenti non vengono modificati.
6. (Facoltativo) Se hai implementato Kubernetes Cluster Autoscaler nel cluster prima di aggiornarlo, aggiorna Cluster Autoscaler alla versione più recente che corrisponde alla versione principale e secondaria di Kubernetes per cui è stato eseguito l'aggiornamento.

   a. Apri la pagina delle versioni di Cluster Autoscaler in un browser Web e trova la versione più recente di Cluster Autoscaler corrispondente alla versione principale e secondaria di Kubernetes del cluster. Ad esempio, se la versione di Kubernetes del cluster è `1.30`, cerca la versione più recente di Cluster Autoscaler che inizia con `1.30`. Registra il numero di versione semantica (`1.30.n`, ad esempio) per tale versione da utilizzare nella fase successiva.

   b. Impostare il tag image di Cluster Autoscaler sulla versione registrata nella fase precedente mediante il comando seguente. Se necessario, sostituire *`1.30.n`* con il valore in proprio possesso.

   ```
   kubectl -n kube-system set image deployment.apps/cluster-autoscaler cluster-autoscaler=registry.k8s.io/autoscaling/cluster-autoscaler:v1.30.n
   ```

7. (Cluster con soli nodi GPU) Se il cluster ha gruppi di nodi con supporto GPU (ad esempio, `p3.2xlarge`), dovrai necessario aggiornare il DaemonSet del plug-in del dispositivo NVIDIA per Kubernetes sul cluster con il seguente comando. Sostituisci*`vX.X.X`*con i tuoi desideriPlugin per dispositivi NVIDIA/K8Sversione prima di eseguire il seguente comando.

   ```
   kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/nvidia-device-plugin.yml
   ```

8. Aggiorna Amazon VPC CNI plugin for Kubernetes, CoreDNS e i componenti aggiuntivi `kube-proxy`. È preferibile aggiornare i componenti aggiuntivi alle versioni minime elencate nei Token dell'account di servizio.

   - Se stai utilizzando i componenti aggiuntivi di Amazon EKS, nella console Amazon EKS seleziona Clusters (Cluster), quindi seleziona il nome del cluster aggiornato nel riquadro di navigazione a sinistra. Le notifiche vengono visualizzate nella console Ti viene segnalato che è disponibile una nuova versione per ogni componente aggiuntivo per cui è disponibile

un aggiornamento. Per aggiornare un componente aggiuntivo, seleziona la scheda Add-ons (Componenti aggiuntivi). In una delle caselle relative al componente aggiuntivo che dispone di un aggiornamento disponibile, selezionare Aggiorna ora, selezionare una versione disponibile e quindi selezionare Aggiorna.

- In alternativa, puoi utilizzare AWS CLI o per `eksctl` aggiornare i componenti aggiuntivi. Per ulteriori informazioni, consulta Aggiornamento di un componente aggiuntivo.

9. Se necessario, aggiorna la tua versione di `kubectl`. Utilizza la versione secondaria `kubectl` immediatamente precedente a quella del piano di controllo del cluster Amazon EKS. Ad esempio, un client `kubectl` `1.29` funziona con i cluster Kubernetes `1.28`, `1.29` e `1.30`. Puoi controllare la versione attualmente installata con il seguente comando.

```
kubectl version --client
```

Eliminazione di un cluster Amazon EKS

Terminato l'utilizzo del cluster Amazon EKS, eliminare le risorse ad esso associate per non dover sostenere costi superflui.

Per rimuovere un cluster connesso, consultare Annullamento della registrazione di un cluster

> **Important**
>
> - Se disponi di servizi attivi nel cluster che sono associati a un load balancer, devi eliminare questi servizi prima di eliminare il cluster per una corretta eliminazione dei load balancer. In caso contrario, potresti avere risorse orfane nel VPC che ti impediscono di eliminarlo.
> - Se viene visualizzato un errore in seguito alla rimozione del creatore del cluster, consultare questo articolo per la risoluzione.
> - Le risorse di Amazon Managed Service for Prometheus non rientrano nel ciclo di vita del cluster e devono essere gestite indipendentemente dal cluster. Quando elimini il cluster, assicurati di eliminare anche tutti gli scraper applicabili per bloccare i costi applicabili. Per ulteriori informazioni, consulta Trova ed elimina gli scraper nella Guida per l'utente di Amazon Managed Service for Prometheus.

Puoi eliminare un cluster con`eksctl`, il, o il AWS Management Console. AWS CLI

eksctl

Eliminare un cluster Amazon EKS e nodi con **`eksctl`**

Questa procedura richiede `eksctl` versione `0.183.0` o successiva. Puoi verificare la versione con il comando seguente:

```
eksctl version
```

Per istruzioni sull'installazione o sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

1. Elenca tutti i servizi in esecuzione nel cluster.

   ```
   kubectl get svc --all-namespaces
   ```

2. Elimina i servizi che hanno un valore `EXTERNAL-IP` associato. Questi servizi sono anticipati da un load balancer Elastic Load Balancing e, per consentire al sistema e alle risorse associate di essere rilasciate correttamente, è necessario eliminarli in Kubernetes.

   ```
   kubectl delete svc service-name
   ```

3. Elimina il cluster e i relativi nodi associati con il comando seguente, sostituendo *`prod`* con il nome del cluster.

   ```
   eksctl delete cluster --name prod
   ```

   Output:

   ```
   [#]  using region region-code
   [#]  deleting EKS cluster "prod"
   [#]  will delete stack "eksctl-prod-nodegroup-standard-nodes"
   [#]  waiting for stack "eksctl-prod-nodegroup-standard-nodes" to get deleted
   [#]  will delete stack "eksctl-prod-cluster"
   [#]  the following EKS cluster resource(s) for "prod" will be deleted: cluster.
    If in doubt, check CloudFormation console
   ```

AWS Management Console

Per eliminare un cluster Amazon EKS con AWS Management Console

1. Elenca tutti i servizi in esecuzione nel cluster.

```
kubectl get svc --all-namespaces
```

2. Elimina i servizi che hanno un valore `EXTERNAL-IP` associato. Questi servizi sono anticipati da un load balancer Elastic Load Balancing e, per consentire al sistema e alle risorse associate di essere rilasciate correttamente, è necessario eliminarli in Kubernetes.

```
kubectl delete svc service-name
```

3. Eliminazione di tutti i gruppi di nodi e profili Fargate.

   a. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.

   b. Nel pannello di navigazione a sinistra, scegli Clusters (Cluster) Amazon EKS, quindi nell'elenco a schede dei cluster scegli il nome del cluster da eliminare.

   c. Seleziona la scheda Compute (Calcolo), quindi scegli un gruppo di nodi da eliminare. Scegli Delete (Elimina), immetti il nome del gruppo di nodi, quindi seleziona Delete (Elimina). Eliminare tutti i gruppi di nodi del cluster.

   > **Note**
   >
   > L'elenco presenta solo gruppi di nodi gestiti.

   d. Scegli un profilo Fargate da eliminare, seleziona Delete (Elimina), immetti il nome del profilo e infine scegli Delete (Elimina). Eliminare tutti i profili di Fargate nel cluster.

4. Elimina tutti gli stack di nodi AWS CloudFormation autogestiti.

   a. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.

   b. Scegli lo stack del nodo da eliminare, quindi scegli Elimina.

   c. Nella finestra di dialogo di conferma Delete stack (Elimina stack) scegliere Delete stack. (Elimina stack). Eliminare tutte le pile di nodi autogestiti nel cluster.

5. Eliminare il cluster.

a. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.

b. Seleziona il cluster da eliminare e scegli Delete (Elimina).

c. Nella schermata di conferma dell'eliminazione del cluster, scegliere Elimina.

6. (Facoltativo) Eliminare lo stack VPC. AWS CloudFormation

a. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.

b. Selezionare lo stack del VPC da eliminare, quindi scegliere Delete (Elimina).

c. Nella finestra di dialogo di conferma Delete stack (Elimina stack) scegliere Delete stack. (Elimina stack).

AWS CLI

Per eliminare un cluster Amazon EKS con AWS CLI

1. Elenca tutti i servizi in esecuzione nel cluster.

```
kubectl get svc --all-namespaces
```

2. Elimina i servizi che hanno un valore `EXTERNAL-IP` associato. Questi servizi sono anticipati da un load balancer Elastic Load Balancing e, per consentire al sistema e alle risorse associate di essere rilasciate correttamente, è necessario eliminarli in Kubernetes.

```
kubectl delete svc service-name
```

3. Eliminazione di tutti i gruppi di nodi e profili Fargate.

a. Elencare i gruppi di nodi nel cluster con il comando seguente.

```
aws eks list-nodegroups --cluster-name my-cluster
```

> **Note**
>
> L'elenco presenta solo gruppi di nodi gestiti.

b. Eliminare ogni gruppo di nodi con il comando seguente. Eliminare tutti i gruppi di nodi del cluster.

```
aws eks delete-nodegroup --nodegroup-name my-nodegroup --cluster-name my-cluster
```

c. Elenca i proqfili Fargate nel cluster con il comando seguente.

```
aws eks list-fargate-profiles --cluster-name my-cluster
```

d. Eliminare ogni profilo di Fargate con il comando seguente. Eliminare tutti i profili di Fargate nel cluster.

```
aws eks delete-fargate-profile --fargate-profile-name my-fargate-profile --cluster-name my-cluster
```

4. Elimina tutti gli stack di nodi AWS CloudFormation autogestiti.

a. Elenca gli AWS CloudFormation stack disponibili con il seguente comando. Trovare il nome del modello del nodo nell'output risultante.

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

b. Elimina lo stack di ogni nodo con il seguente comando, sostituendo *node-stack* con il nome del tuo stack. Eliminare tutte le pile di nodi autogestiti nel cluster.

```
aws cloudformation delete-stack --stack-name node-stack
```

5. Elimina il cluster con il seguente comando, sostituendo *my-cluster* con il nome del tuo cluster.

```
aws eks delete-cluster --name my-cluster
```

6. (Facoltativo) Eliminare lo stack VPC. AWS CloudFormation

a. Elenca gli AWS CloudFormation stack disponibili con il seguente comando. Trovare il nome del modello di VPC nell'output risultante.

```
aws cloudformation list-stacks --query "StackSummaries[].StackName"
```

b. Elimina lo stack VPC con il seguente comando, sostituendo *my-vpc-stack* con il nome dello stack VPC.

```
aws cloudformation delete-stack --stack-name my-vpc-stack
```

Controllo accessi all'endpoint del cluster Amazon EKS

In questa sezione viene descritto come abilitare l'accesso privato per l'endpoint del server API Kubernetesdel cluster Amazon EKS e limitare, o disabilitare completamente, l'accesso pubblico da Internet.

Quando si crea un nuovo cluster, Amazon EKS crea un endpoint per il server API Kubernetes gestito utilizzato per comunicare con il cluster (usando strumenti di gestione Kubernetes, ad esempio `kubectl`). Per impostazione predefinita, questo endpoint del server API è pubblico su Internet e l'accesso al server API è protetto utilizzando una combinazione di AWS Identity and Access Management (IAM) e Role Kubernetes Based Access Control (RBAC) nativo.

Puoi abilitare l'accesso privato al server API Kubernetes in modo che tutte le comunicazioni tra i nodi e il server API rimangano all'interno del VPC. È possibile limitare gli indirizzi IP che possono accedere al server API da Internet o disabilitare completamente l'accesso a Internet al server API.

> **Note**
>
> Poiché questo endpoint è destinato al server Kubernetes API e non a un AWS PrivateLink endpoint tradizionale per la comunicazione con un' AWS API, non viene visualizzato come endpoint nella console Amazon VPC.

Quando si abilita l'accesso privato all'endpoint per il cluster, Amazon EKS crea una zona ospitata privata Route 53 per conto dell'utente e la associa al VPC del cluster. Questa zona ospitata privata è gestita da Amazon EKS e non viene visualizzata nelle risorse Route 53 dell'account. Affinché la zona ospitata privata instradi correttamente il traffico verso il tuo server API, il VPC deve avere `enableDnsHostnames` e `enableDnsSupport` impostati su `true` e le opzioni DHCP impostate per il VPC devono includere `AmazonProvidedDNS` nell'elenco dei server dei nomi di dominio. Per ulteriori informazioni, consultare Visualizzazione e aggiornamento del supporto DNS per il VPC nella Guida per l'utente di Amazon VPC.

È possibile definire i requisiti di accesso all'endpoint del server API quando si crea un nuovo cluster e aggiornare l'accesso endpoint del server API per un cluster in qualsiasi momento.

Modifica dell'accesso all'endpoint del cluster

Utilizza le procedure in questa sezione per modificare l'accesso all'endpoint per un cluster esistente. La tabella seguente mostra le combinazioni di accesso all'endpoint del server API supportate e il comportamento associato.

Opzioni di accesso all'endpoint del server API

Accesso pubblico all'endpoint	Accesso privato all'endpoint	Comportamento
Abilitato	Disabilitato	• Questo è il comportamento di default per nuovi cluster Amazon EKS. • Le richieste API Kubernetes provenienti dal VPC del cluster (ad esempio, la comunicazione tra nodo e piano di controllo) lasciano il VPC ma non la rete Amazon. • Il server API del cluster è accessibile da Internet. È possibile, facoltativamente, limitare i blocchi CIDR che possono accedere all'endpoint pubblico. Se limiti l'accesso a blocchi CIDR specifici, è consigliabile abilitare anche l'endpoint privato o assicurarsi che i blocchi CIDR specificati includano gli indirizzi da cui i nodi e i Pods Fargate

Accesso pubblico all'endpoint	Accesso privato all'endpoint	Comportamento
		(se utilizzati) accedono all'endpoint pubblico.
Abilitato	Abilitato	• Le richieste API Kubernetes all'interno del VPC del cluster (ad esempio, la comunicazione tra nodo e piano di controllo) utilizzano l'endpoint VPC privato. • Il server API del cluster è accessibile da Internet. È possibile, facoltativamente, limitare i blocchi CIDR che possono accedere all'endpoint pubblico.

Accesso pubblico all'endpoint	Accesso privato all'endpoint	Comportamento
Disabilitato	Abilitato	• Tutto il traffico verso il server API del cluster deve provenire dal VPC del cluster o da una rete connessa. • Non esiste alcun accesso pubblico al server API da Internet. Tutti i comandi `kubectl` devono provenire dall'interno del VPC o da una rete connessa. Per le opzioni di connettività, consultare Accesso a un server API solo privato. • L'endpoint del server API del cluster viene risolto dai server DNS pubblici a un indirizzo IP privato dal VPC. In passato, l'endpoint poteva essere risolto solo dall'interno del VPC. Se l'endpoint non viene risolto in un indirizzo IP privato all'interno del VPC per un cluster esistente, è possibile: • Abilitare l'accesso pubblico e quindi disabilitarlo di nuovo. È necessario farlo solo una volta per un cluster e l'endpoint si risolverà in un indirizzo IP

Accesso pubblico all'endpoint	Accesso privato all'endpoint	Comportamento
		privato da quel punto in avanti. • Aggiornare il cluster.

Puoi modificare l'accesso agli endpoint del server API del cluster utilizzando o. AWS Management Console AWS CLI

AWS Management Console

Per modificare l'accesso agli endpoint del server API del cluster utilizzando il AWS Management Console

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegliere il nome del cluster per visualizzare le informazioni sul cluster.
3. Scegliere la scheda Reti, quindi Aggiorna.
4. Per Private access (Accesso privato) scegli se attivare o disattivare l'acceso privato per l'endpoint del server API Kubernetes del cluster. Se abiliti l'accesso privato, le richieste API Kubernetes che provengono dal VPC del cluster utilizzeranno l'endpoint VPC privato. Per disabilitare l'accesso pubblico è necessario abilitare l'accesso privato.
5. Per Public access (Accesso pubblico) scegli se attivare o disattivare l'acceso pubblico per l'endpoint del server API Kubernetes del cluster. Se disabiliti l'accesso pubblico, il server API Kubernetes del cluster potrà ricevere richieste solo dal VPC del cluster.
6. (Facoltativo) Se è stato abilitato Accesso pubblic, è possibile specificare quali indirizzi Internet possono comunicare con l'endpoint pubblico. Selezionare Advanced settings (Impostazioni avanzate). Immettere un blocco CIDR, ad esempio *203.0.113.5/32*. Il blocco non può includere indirizzi riservati. È possibile immettere blocchi aggiuntivi selezionando Add Source (Aggiungi origine). È possibile specificare un numero massimo di blocchi CIDR. Per ulteriori informazioni, consulta Service Quotas di Amazon EKS. Se non si specificano blocchi, l'endpoint del server API pubblico riceve richieste da tutti gli indirizzi IP (`0.0.0.0/0`). Se limiti l'accesso all'endpoint pubblico utilizzando i blocchi CIDR, è consigliabile abilitare anche l'accesso agli endpoint privati in modo che i nodi e i Pods Fargate (se utilizzati) possano comunicare con il cluster. Senza l'endpoint privato abilitato, le origini CIDR dell'endpoint di accesso pubblico devono includere le origini di uscita dal VPC. Ad esempio, se si dispone

di un nodo in una sottorete privata che comunica su Internet tramite un gateway NAT, sarà necessario aggiungere l'indirizzo IP in uscita del gateway NAT come parte di un blocco CIDR nella whitelist nell'endpoint pubblico.

7. Scegliere Update (Aggiorna) per terminare.

AWS CLI

Per modificare l'accesso all'endpoint del server API del cluster con la AWS CLI

Completa i seguenti passaggi utilizzando la AWS CLI versione `1.27.160` o successiva. È possibile verificare la versione corrente con `aws --version`. Per installare o aggiornare il AWS CLI, vedere Installazione di AWS CLI.

1. Aggiornare l'accesso all'endpoint del server API del cluster con il comando AWS CLI seguente. Sostituire il nome del cluster e i valori di accesso dell'endpoint desiderati. Se si imposta `endpointPublicAccess=true`, è possibile (facoltativamente) immettere un singolo blocco CIDR o un elenco separato da virgole di blocchi CIDR per `publicAccessCidrs`. I blocchi non possono includere indirizzi riservati. Se si specificano blocchi CIDR, l'endpoint del server API pubblico riceverà solo le richieste dai blocchi elencati. È possibile specificare un numero massimo di blocchi CIDR. Per ulteriori informazioni, consulta Service Quotas di Amazon EKS. Se limiti l'accesso all'endpoint pubblico utilizzando i blocchi CIDR, è consigliabile abilitare anche l'accesso agli endpoint privati in modo che i nodi e i Pods Fargate (se utilizzati) possano comunicare con il cluster. Senza l'endpoint privato abilitato, le origini CIDR dell'endpoint di accesso pubblico devono includere le origini di uscita dal VPC. Ad esempio, se si dispone di un nodo in una sottorete privata che comunica su Internet tramite un gateway NAT, sarà necessario aggiungere l'indirizzo IP in uscita del gateway NAT come parte di un blocco CIDR nella whitelist nell'endpoint pubblico. Se non si specificano blocchi CIDR, l'endpoint del server API pubblico riceve richieste da tutti gli indirizzi IP (0.0.0.0/0).

> **Note**
>
> Il comando seguente consente l'accesso privato e l'accesso pubblico da un singolo indirizzo IP per l'endpoint del server API. Sostituire *`203.0.113.5/32`* con un singolo blocco CIDR o un elenco separato da virgole di blocchi CIDR a cui si desidera limitare l'accesso alla rete.

```
aws eks update-cluster-config \
    --region region-code \
    --name my-cluster \
    --resources-vpc-config
 endpointPublicAccess=true,publicAccessCidrs="203.0.113.5/32",endpointPrivateAccess=true
```

Di seguito viene riportato un output di esempio:

```
{
    "update": {
        "id": "e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000",
        "status": "InProgress",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "true"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            },
            {
                "type": "publicAccessCidrs",
                "value": "[\203.0.113.5/32\"]"
            }
        ],
        "createdAt": 1576874258.137,
        "errors": []
    }
}
```

2. Monitorare lo stato di aggiornamento dell'accesso all'endpoint con il comando seguente, utilizzando il nome del cluster, e aggiornare l'ID restituito dal comando precedente. L'aggiornamento è completo quando lo stato è illustrato come `Successful`.

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000
```

Di seguito viene riportato un output di esempio:

```
{
    "update": {
        "id": "e6f0905f-a5d4-4a2a-8c49-EXAMPLE00000",
        "status": "Successful",
        "type": "EndpointAccessUpdate",
        "params": [
            {
                "type": "EndpointPublicAccess",
                "value": "true"
            },
            {
                "type": "EndpointPrivateAccess",
                "value": "true"
            },
            {
                "type": "publicAccessCidrs",
                "value": "[\203.0.113.5/32\"]"
            }
        ],
        "createdAt": 1576874258.137,
        "errors": []
    }
}
```

Accesso a un server API solo privato

Se hai disabilitato l'accesso pubblico per l'endpoint del server API Kubernetes del cluster, potrai accedere al server API solo dal VPC o da una rete connessa. Di seguito sono elencati alcuni possibili modi per accedere all'endpoint del server API Kubernetes:

Rete connessa

È possibile connettere la rete al VPC con un gateway di transito AWS o un'altra opzione di connettività e quindi utilizzare un computer nella rete connessa. Il gruppo di sicurezza del piano di controllo di Amazon EKS deve contenere le regole per consentire il traffico in ingresso sulla porta 443 dalla rete connessa.

Host bastion Amazon EC2

È possibile avviare un'istanza Amazon EC2 in una sottorete pubblica nel VPC del cluster e quindi accedere tramite SSH a tale istanza per eseguire comandi `kubectl`. Per ulteriori informazioni, consulta Host bastione Linux su AWS. Il gruppo di sicurezza del piano di controllo di Amazon EKS deve contenere le regole per consentire il traffico in ingresso sulla porta 443 dal bastion host. Per ulteriori informazioni, consulta Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS.

Quando configuri `kubectl` per l'host bastione, assicurati di utilizzare le credenziali AWS che sono già associate alla configurazione RBAC del cluster o aggiungere il principale IAM che verrà utilizzato dall'host bastione nella configurazione RBAC prima di rimuovere l'accesso pubblico per l'endpoint. Per ulteriori informazioni, consulta the section called "Concedi l'accesso alle API Kubernetes" e Accesso negato o non autorizzato (`kubectl`).

AWS Cloud9 IDE

AWS Cloud9 è un ambiente di sviluppo integrato (IDE) basato su cloud che consente di scrivere, eseguire ed eseguire il debug del codice con un semplice browser. È possibile creare un AWS Cloud9 IDE nel VPC del cluster e utilizzare l'IDE per comunicare con il cluster. Per ulteriori informazioni, consultare Creazione di un ambiente in AWS Cloud9. È necessario assicurarsi che il gruppo di sicurezza del piano di controllo Amazon EKS contenga regole per consentire il traffico in ingresso sulla porta 443 dal gruppo di sicurezza IDE. Per ulteriori informazioni, consulta Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS.

Quando esegui la configurazione `kubectl` per il tuo AWS Cloud9 IDE, assicurati di utilizzare AWS credenziali già mappate alla configurazione RBAC del cluster oppure aggiungi il principio IAM che l'IDE utilizzerà alla configurazione RBAC prima di rimuovere l'accesso pubblico all'endpoint. Per ulteriori informazioni, consultare Concedi l'accesso alle Kubernetes API e Accesso negato o non autorizzato (`kubectl`).

Abilitazione della crittografia segreta dei dati in transito su un cluster esistente

Se abiliti la crittografia dei segreti, i segreti di Kubernetes vengono crittografati utilizzando la AWS KMS key selezionata. La chiave KMS deve soddisfare le condizioni seguenti:

- Simmetria

- Possibilità di crittografia e decrittografia dei dati
- Creazione nella stessa Regione AWS del cluster
- Se la chiave KMS è stata creata in un account diverso, il principale IAM dovrà avere accesso alla stessa.

Per ulteriori informazioni, consulta Autorizzazione per i principali IAM in altri account di utilizzare una chiave KMS nella Guida per gli sviluppatori di AWS Key Management Service.

> **Warning**
>
> Non è possibile disabilitare la crittografia dei segreti dopo averla abilitata. Questa operazione è irreversibile.

eksctl

È possibile abilitare la crittografia in due modi:

- Aggiungere la crittografia al cluster con un singolo comando.

  Per ricrittografare automaticamente i segreti, esegui il comando seguente.

  ```
  eksctl utils enable-secrets-encryption \
      --cluster my-cluster \
      --key-arn arn:aws:kms:region-code:account:key/key
  ```

  Per disattivare la ricrittografia automatica dei segreti, esegui il comando seguente.

  ```
  eksctl utils enable-secrets-encryption
      --cluster my-cluster \
      --key-arn arn:aws:kms:region-code:account:key/key \
      --encrypt-existing-secrets=false
  ```

- Aggiungi la crittografia al cluster con un file `kms-cluster.yaml`.

  ```
  apiVersion: eksctl.io/v1alpha5
  kind: ClusterConfig

  metadata:
    name: my-cluster
  ```

```
  region: region-code

secretsEncryption:
  keyARN: arn:aws:kms:region-code:account:key/key
```

Per ricrittografare automaticamente i segreti, esegui il comando seguente.

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml
```

Per disattivare la ricrittografia automatica dei segreti, esegui il comando seguente.

```
eksctl utils enable-secrets-encryption -f kms-cluster.yaml --encrypt-existing-
secrets=false
```

AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli il cluster a cui aggiungere la crittografia KMS.
3. Scegli la scheda Overview (Panoramica) (selezionata per impostazione predefinita).
4. Scorri verso il basso fino a Secrets encryption (Crittografia dei segreti) e scegli Enable (Abilita).
5. Seleziona una chiave nell'elenco a discesa e fare clic sul pulsante Enable (Abilita). Se non sono elencate chiavi, è necessario crearne una prima. Per ulteriori informazioni, consultare Creazione chiavi
6. Fai clic sul pulsante Confirm (Conferma) per utilizzare la chiave scelta.

AWS CLI

1. Associa la configurazione di crittografia dei segreti con il cluster utilizzando il comando AWS CLI. Sostituisci i *example values* con i valori in tuo possesso.

```
aws eks associate-encryption-config \
    --cluster-name my-cluster \
    --encryption-config '[{"resources":["secrets"],"provider":
{"keyArn":"arn:aws:kms:region-code:account:key/key"}}]'
```

Di seguito viene riportato un output di esempio:

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "InProgress",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
      }
    ],
    "createdAt": 1613754188.734,
    "errors": []
  }
}
```

2. È possibile monitorare lo stato dell'aggiornamento di crittografia con il comando seguente. Usa il `cluster name` specifico e il valore `update ID` restituito nell'output precedente. Quando viene visualizzato lo stato `Successful`, l'aggiornamento è completo.

```
aws eks describe-update \
    --region region-code \
    --name my-cluster \
    --update-id 3141b835-8103-423a-8e68-12c2521ffa4d
```

Di seguito viene riportato un output di esempio:

```
{
  "update": {
    "id": "3141b835-8103-423a-8e68-12c2521ffa4d",
    "status": "Successful",
    "type": "AssociateEncryptionConfig",
    "params": [
      {
        "type": "EncryptionConfig",
        "value": "[{\"resources\":[\"secrets\"],\"provider\":{\"keyArn\":
\"arn:aws:kms:region-code:account:key/key\"}}]"
      }
```

```
    ],
    "createdAt": 1613754188.734>,
    "errors": []
  }
}
```

3. Per verificare che la crittografia sia attivata nel cluster, eseguire il comando `describe-cluster`. La risposta contiene una stringa `EncryptionConfig`.

```
aws eks describe-cluster --region region-code --name my-cluster
```

Dopo aver abilitato la crittografia nel cluster, sarà necessario crittografare tutti i segreti esistenti con la nuova chiave:

> **Note**
>
> È necessario eseguire il comando seguente soltanto se si utilizza `eksctl` e la crittografia automatica dei segreti è disattivata.

```
kubectl get secrets --all-namespaces -o json | kubectl annotate --overwrite -f - kms-encryption-timestamp="time value"
```

> **Warning**
>
> Se si abilita la crittografia dei segreti per un cluster esistente e la chiave KMS utilizzata non viene mai eliminata, non sarà possibile ripristinare il cluster. L'eliminazione della chiave KMS metterà definitivamente il cluster in uno stato degradato. Per ulteriori informazioni, consultare Eliminazione di chiavi KMS AWS.

> **Note**
>
> Per impostazione predefinita, il comando `create-key` crea una chiave KMS con crittografia simmetrica con una policy della chiave che consente all'amministratore principale dell'account di accedere alle azioni e alle risorse AWS KMS. Se si desidera ridurre l'ambito delle autorizzazioni, assicurarsi che le azioni `kms:DescribeKey` e `kms:CreateGrant` siano consentite nella policy per il principale che effettua la chiamata all'API `create-cluster`.

Per cluster che utilizzano la crittografia a busta KMS, sono necessarie le autorizzazioni `kms:CreateGrant`. La condizione non `kms:GrantIsForAWSResource` è supportata per l' CreateClusterazione e non deve essere utilizzata nelle politiche KMS per controllare le `kms:CreateGrant` autorizzazioni degli utenti che eseguono. CreateCluster

Abilitazione del supporto di Windows per il cluster Amazon EKS

Prima di implementare i nodi di Windows, tieni conto delle seguenti considerazioni.

Considerazioni

- Puoi utilizzare la rete host sui nodi Windows utilizzando i Pod `HostProcess` . Per ulteriori informazioni, consulta Creazione di `HostProcess`Pod per Windows nella documentazione di Kubernetes.
- I cluster Amazon EKS devono contenere uno o più nodi Linux o Fargate per eseguire i Pods del sistema core che funzionano solo su Linux, come CoreDNS.
- I log degli eventi `kubelet` e `kube-proxy` vengono reindirizzati al log degli eventi EKS Windows e sono impostati su un limite di 200 MB.
- Non puoi utilizzare Gruppi di sicurezza per Pods con Pods in esecuzione sui nodi Windows.
- Non puoi utilizzare una rete personalizzata con i nodi Windows.
- Non puoi utilizzare `IPv6` con nodi di Windows.
- I nodi di Windows supportano un'interfaccia di rete elastica per nodo. Per impostazione predefinita, il numero di Pods che puoi eseguire per ogni nodo di Windows è uguale al numero di indirizzi IP meno uno disponibili per ogni interfaccia di rete elastica relativa al tipo di istanza del nodo. Per ulteriori informazioni, consulta Indirizzi IP per interfaccia di rete per tipo di istanza nella Guida per l'utente di Amazon EC2.
- In un cluster Amazon EKS, un singolo servizio con un load balancer può supportare fino a 1024 Pods di back-end. Ogni Pod ha il proprio indirizzo IP univoco. Il limite precedente di 64 Pods non è più valido, dopo un aggiornamento di Windows Server a partire dalla Build del sistema operativo 17763.2746.
- I container Windows non sono supportati per i Pods Amazon EKS su Fargate.
- Non è possibile recuperare i log dal pod `vpc-resource-controller`. In precedenza era possibile quando si implementava il controller sul piano dati.

- Esiste un periodo di raffreddamento prima che un indirizzo `IPv4` venga assegnato a un nuovo pod. In questo modo, si impedisce che il traffico vada verso un pod precedente con lo stesso indirizzo `IPv4` a causa della mancata validità delle regole `kube-proxy`.
- L'origine per il controller è gestita su GitHub. Per contribuire o segnalare problemi relativi al controller, consulta il progetto su GitHub.
- Quando specifichi un ID AMI personalizzato per i gruppi di nodi Windows gestiti, aggiungilo `eks:kube-proxy-windows` alla mappa di configurazione di AWS IAM Authenticator. Per ulteriori informazioni, consulta Limiti e condizioni quando si specifica un ID AMI.

Prerequisiti

- Un cluster esistente. Il cluster deve eseguire una delle versioni Kubernetes e della piattaforma elencate nella tabella seguente. Sono supportate anche tutte le versioni Kubernetes e della piattaforma successive a quelle elencate. Se la versione del cluster o della piattaforma è precedente a una delle seguenti versioni, dovrai abilitare il supporto Windows legacy sul piano dati del cluster. Una volta che il cluster si trova in una delle seguenti versioni Kubernetes e della piattaforma o versioni successive, puoi rimuovere il supporto Windows legacy e abilitare il supporto Windows sul piano di controllo.

Versione Kubernetes	Versione della piattaforma
1.30	eks.2
1,29	eks.1
1,28	eks.1
1,27	eks.1
1,26	eks.1
1,25	eks.1
1,24	eks.2

- Il tuo cluster deve avere almeno un nodo Linux (ne consigliamo almeno due) o un Pod Fargate per eseguire CoreDNS. Se abiliti il supporto Windows legacy, devi utilizzare un nodo Linux (non è possibile utilizzare un Pod Fargate) per eseguire CoreDNS.

- Un Ruolo IAM del cluster Amazon EKS esistente.

Abilitazione del supporto Windows

Se la versione del cluster non corrisponde a una delle versioni o versioni successive della piattaforma e di Kubernetes elencate in Prerequisiti, devi abilitare invece il supporto Windows legacy. Per ulteriori informazioni, consulta Abilitazione del supporto Windows legacy.

Se non hai mai abilitato il supporto Windows sul cluster, vai al passaggio successivo.

Se hai abilitato il supporto Windows su un cluster precedente a una versione Kubernetes o della piattaforma elencata nei Prerequisiti, per prima cosa rimuovi `vpc-resource-controller` e `vpc-admission-webhook` dal tuo piano dati. Sono obsoleti e non sono più necessari.

Abilitazione del supporto Windows per il cluster

1. Se non disponi di nodi Amazon Linux nel cluster e utilizzi gruppi di sicurezza per i Pods, vai al passaggio successivo. Altrimenti, conferma che la policy gestita di `AmazonEKSVPCResourceController` è collegata al tuo ruolo del cluster. Sostituisci *`eksClusterRole`* con il nome del ruolo del cluster.

   ```
   aws iam list-attached-role-policies --role-name eksClusterRole
   ```

   Di seguito viene riportato un output di esempio:

   ```
   {
       "AttachedPolicies": [
           {
               "PolicyName": "AmazonEKSClusterPolicy",
               "PolicyArn": "arn:aws:iam::aws:policy/AmazonEKSClusterPolicy"
           },
           {
               "PolicyName": "AmazonEKSVPCResourceController",
               "PolicyArn": "arn:aws:iam::aws:policy/AmazonEKSVPCResourceController"
           }
       ]
   }
   ```

   Se la policy è collegata, come nell'output precedente, salta il passaggio successivo.

2. Allega la politica gestita da AmazonekSVPC ResourceController al tuo. Ruolo IAM del cluster Amazon EKS Sostituisci *eksClusterRole* con il nome del ruolo del cluster.

```
aws iam attach-role-policy \
  --role-name eksClusterRole \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
```

3. Crea un file denominato *vpc-resource-controller-configmap.yaml* con i seguenti contenuti.

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: amazon-vpc-cni
  namespace: kube-system
data:
  enable-windows-ipam: "true"
```

4. Applica la `ConfigMap` al cluster.

```
kubectl apply -f vpc-resource-controller-configmap.yaml
```

5. Verifica che `aws-auth` `ConfigMap` contenga una mappatura per il ruolo di istanza del nodo Windows per includere il gruppo di autorizzazioni `eks:kube-proxy-windows` RBAC. Puoi eseguire la verifica eseguendo il comando seguente.

```
kubectl get configmap aws-auth -n kube-system -o yaml
```

Di seguito viene riportato un output di esempio:

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: aws-auth
  namespace: kube-system
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
```

```
      - eks:kube-proxy-windows # This group is required for Windows DNS resolution
 to work
      rolearn: arn:aws:iam::111122223333:role/eksNodeRole
      username: system:node:{{EC2PrivateDNSName}}
[...]
```

Dovresti vedere `eks:kube-proxy-windows` elencato nei gruppi. Se il gruppo non è specificato, devi aggiornare la `ConfigMap` o crearla per includere il gruppo richiesto. Per ulteriori informazioni su `aws-auth` `ConfigMap`, consulta Applica la `aws-auth` `ConfigMap` al cluster.

Rimozione del supporto Windows legacy dal piano dati

Se hai abilitato il supporto Windows su un cluster precedente a una versione Kubernetes o della piattaforma elencata nei Prerequisiti, per prima cosa rimuovi `vpc-resource-controller` e `vpc-admission-webhook` dal tuo piano dati. Sono obsoleti e non più necessari in quanto la funzionalità fornita è ora abilitata sul piano di controllo.

1. Disinstallare il `vpc-resource-controller` con il seguente comando. Utilizza questo comando indipendentemente dallo strumento con cui è stato originariamente installato. Sostituisci *`region-code`* (solo l'istanza del testo dopo `/manifests/`) con la Regione AWS in cui si trova il cluster.

   ```
   kubectl delete -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
   ```

2. Disinstalla il `vpc-admission-webhook` seguendo le istruzioni dello strumento con cui è stato installato.

   eksctl

   Esegui i comandi seguenti.

   ```
   kubectl delete deployment -n kube-system vpc-admission-webhook
   kubectl delete service -n kube-system vpc-admission-webhook
   kubectl delete mutatingwebhookconfigurations.admissionregistration.k8s.io vpc-admission-webhook-cfg
   ```

kubectl on macOS or Windows

Esegui il comando seguente. Sostituisci *region-code* (solo l'istanza di quel testo dopo/`manifests/`) con quello in Regione AWS cui si trova il cluster.

```
kubectl delete -f https://s3.us-west-2.amazonaws.com/amazon-
eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-
deployment.yaml
```

3. Abilita il supporto Windows per il tuo cluster sul piano di controllo.

Disabilitazione del supporto Windows

Disabilitazione del supporto Windows sul cluster

1. Se il cluster contiene nodi Amazon Linux e utilizzi gruppi di sicurezza per pod Pods, salta questo passaggio.

   Rimuovi la policy IAM gestita `AmazonVPCResourceController` dal ruolo del cluster. Sostituisci *eksClusterRole* con il nome del ruolo del tuo cluster e *111122223333* con il tuo ID account.

   ```
   aws iam detach-role-policy \
       --role-name eksClusterRole \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKSVPCResourceController
   ```

2. Disabilita Windows IPAM in. `amazon-vpc-cni` ConfigMap

   ```
   kubectl patch configmap/amazon-vpc-cni \
                       -n kube-system \
                       --type merge \
                       -p '{"data":{"enable-windows-ipam":"false"}}'
   ```

Implementazione di pod

Quando si implementano i pod nel cluster, è necessario specificare il sistema operativo utilizzato se si esegue una combinazione di tipi di nodi.

Per i Pods Linux, utilizza il seguente testo del selettore dei nodi nei manifesti.

```
nodeSelector:
        kubernetes.io/os: linux
        kubernetes.io/arch: amd64
```

Per i Pods Windows, utilizza il seguente testo del selettore dei nodi nei manifesti.

```
nodeSelector:
        kubernetes.io/os: windows
        kubernetes.io/arch: amd64
```

È possibile implementare un'applicazione di esempio per vedere i selettori dei nodi in uso.

Abilitazione del supporto Windows legacy

Se il cluster non corrisponde a una delle versioni Kubernetes e della piattaforma elencate nei Prerequisiti o a una versione successiva, consigliamo di abilitare il supporto Windows sul piano di controllo. Per ulteriori informazioni, consulta Abilitazione del supporto Windows.

I passaggi seguenti consentono di abilitare il supporto Windows legacy per il piano dati del cluster Amazon EKS se la versione del cluster o della piattaforma è precedente alle versioni elencate nei Prerequisiti. Una volta che la versione del cluster e della piattaforma corrispondono a una versione elencata nei Prerequisiti o a una versione successiva, ti consigliamo di rimuovere il supporto Windows legacy e di abilitarlo per il tuo piano di controllo.

Puoi utilizzare `eksctl`, un client Windows o un client macOS o Linux per abilitare il supporto Windows legacy per il cluster.

eksctl

Abilitazione del supporto Windows legacy per il cluster con **`eksctl`**

Prerequisito

Questa procedura richiede `eksctl` versione `0.183.0` o successiva. È possibile verificare la tua versione con il seguente comando.

```
eksctl version
```

Per ulteriori informazioni sull'aggiornamento o sull'installazione di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

1. Abilita il supporto Windows per il tuo cluster Amazon EKS con il seguente comando `eksctl`. Sostituisci *`my-cluster`* con il nome del cluster. Questo comando distribuisce il controller delle risorse VPC e il webhook del controller di ammissione VPC richiesti sui cluster Amazon EKS per eseguire i carichi di lavoro di Windows.

```
eksctl utils install-vpc-controllers --cluster my-cluster --approve
```

> **Important**
>
> Il webhook del controller di ammissione VPC è firmato con un certificato con scadenza ad un anno dopo la data di rilascio. Per evitare tempi di inattività, assicurarsi di rinnovare il certificato prima della scadenza. Per ulteriori informazioni, consulta Rinnovo del certificato webhook di ammissione VPC.

2. Dopo aver attivato il supporto Windows, potrai avviare un gruppo di nodi Windows nel cluster. Per ulteriori informazioni, consulta Avvio dei nodi Windows autogestiti.

Windows

Abilitazione del supporto Windows legacy per il cluster con un client Windows

Nella procedura seguente, sostituisci *`region-code`* con la Regione AWS in cui si trova il cluster.

1. Implementare il controller di risorse VPC nel cluster.

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
```

2. Implementare il webhook del controller di ammissione VPC nel cluster.

   a. Scaricare gli script e i file di implementazione richiesti.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-deployment.yaml;
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/Setup-VPCAdmissionWebhook.ps1;
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.ps1;
```

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/
vpc-admission-webhook/latest/webhook-patch-ca-bundle.ps1;
```

b. Installare OpenSSL e jq.

c. Configurare e implementare il webhook di ammissione VPC.

```
./Setup-VPCAdmissionWebhook.ps1 -DeploymentTemplate ".\vpc-admission-
webhook-deployment.yaml"
```

> **Important**
>
> Il webhook del controller di ammissione VPC è firmato con un certificato con scadenza ad un anno dopo la data di rilascio. Per evitare tempi di inattività, assicurarsi di rinnovare il certificato prima della scadenza. Per ulteriori informazioni, consulta Rinnovo del certificato webhook di ammissione VPC.

3. Determinare se il cluster dispone dell'associazione di ruolo del cluster richiesta.

```
kubectl get clusterrolebinding eks:kube-proxy-windows
```

Se viene restituito un output simile a quello del seguente esempio, il cluster dispone dell'associazione di ruolo necessaria.

```
NAME                       AGE
eks:kube-proxy-windows     10d
```

Se l'output include `Error from server (NotFound)`, il cluster non dispone dell'associazione di ruolo richiesta. Aggiungere l'associazione creando un file denominato *`eks-kube-proxy-windows-crb.yaml`* con il seguente contenuto.

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: eks:kube-proxy-windows
  labels:
    k8s-app: kube-proxy
    eks.amazonaws.com/component: kube-proxy
subjects:
  - kind: Group
```

```
    name: "eks:kube-proxy-windows"
roleRef:
  kind: ClusterRole
  name: system:node-proxier
  apiGroup: rbac.authorization.k8s.io
```

Applicare la configurazione al cluster.

```
kubectl apply -f eks-kube-proxy-windows-crb.yaml
```

4. Dopo aver attivato il supporto Windows, potrai avviare un gruppo di nodi Windows nel cluster. Per ulteriori informazioni, consulta Avvio dei nodi Windows autogestiti.

macOS and Linux

Abilitazione del supporto Windows legacy per il cluster con un client macOS o Linux

Questa procedura richiede che la libreria `openssl` e il processore JSON `jq` siano installati sul sistema client.

Nella procedura seguente, sostituisci *region-code* con la Regione AWS in cui si trova il cluster.

1. Implementare il controller di risorse VPC nel cluster.

```
kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-resource-controller/latest/vpc-resource-controller.yaml
```

2. Creare il manifesto del webhook del controller di ammissione VPC per il cluster.

   a. Scaricare gli script e i file di implementazione richiesti.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.sh
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-patch-ca-bundle.sh
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/vpc-admission-webhook-deployment.yaml
```

   b. Aggiungere le autorizzazioni agli script della shell in modo che possano essere eseguiti.

```
chmod +x webhook-create-signed-cert.sh webhook-patch-ca-bundle.sh
```

c. Creare un segreto per una comunicazione protetta.

```
./webhook-create-signed-cert.sh
```

d. Verificare il segreto.

```
kubectl get secret -n kube-system vpc-admission-webhook-certs
```

e. Configurare il webhook e creare un file di implementazione.

```
cat ./vpc-admission-webhook-deployment.yaml | ./webhook-patch-ca-bundle.sh >
 vpc-admission-webhook.yaml
```

3. Implementare il webhook di ammissione VPC.

```
kubectl apply -f vpc-admission-webhook.yaml
```

> **Important**
>
> Il webhook del controller di ammissione VPC è firmato con un certificato con scadenza ad un anno dopo la data di rilascio. Per evitare tempi di inattività, assicurarsi di rinnovare il certificato prima della scadenza. Per ulteriori informazioni, consulta Rinnovo del certificato webhook di ammissione VPC.

4. Determinare se il cluster dispone dell'associazione di ruolo del cluster richiesta.

```
kubectl get clusterrolebinding eks:kube-proxy-windows
```

Se viene restituito un output simile a quello del seguente esempio, il cluster dispone dell'associazione di ruolo necessaria.

```
NAME                     ROLE                                AGE
eks:kube-proxy-windows   ClusterRole/system:node-proxier   19h
```

Se l'output include `Error from server (NotFound)`, il cluster non dispone dell'associazione di ruolo richiesta. Aggiungere l'associazione creando un file denominato *`eks-kube-proxy-windows-crb.yaml`* con il seguente contenuto.

```
kind: ClusterRoleBinding
apiVersion: rbac.authorization.k8s.io/v1beta1
metadata:
  name: eks:kube-proxy-windows
  labels:
    k8s-app: kube-proxy
    eks.amazonaws.com/component: kube-proxy
subjects:
  - kind: Group
    name: "eks:kube-proxy-windows"
roleRef:
  kind: ClusterRole
  name: system:node-proxier
  apiGroup: rbac.authorization.k8s.io
```

Applicare la configurazione al cluster.

```
kubectl apply -f eks-kube-proxy-windows-crb.yaml
```

5. Dopo aver attivato il supporto Windows, potrai avviare un gruppo di nodi Windows nel cluster. Per ulteriori informazioni, consulta Avvio dei nodi Windows autogestiti.

Rinnovo del certificato webhook di ammissione VPC

Il certificato utilizzato dal webhook di ammissione VPC scade un anno dopo il rilascio. Per evitare tempi di inattività, è importante rinnovare il certificato prima della scadenza. È possibile verificare la data di scadenza del certificato corrente con il seguente comando.

```
kubectl get secret \
    -n kube-system \
    vpc-admission-webhook-certs -o json | \
    jq -r '.data."cert.pem"' | \
    base64 -decode | \
    openssl x509 \
    -noout \
    -enddate | \
    cut -d= -f2
```

Di seguito viene riportato un output di esempio:

```
May 28 14:23:00 2022 GMT
```

Puoi rinnovare il certificato utilizzando `eksctl` o un computer Windows o un Linux/macOS. Seguire le istruzioni per lo strumento utilizzato originariamente per installare il webhook di ammissione VPC. Ad esempio, se in origine è stato installato il webhook di ammissione VPC utilizzando `eksctl`, è necessario rinnovare il certificato utilizzando le istruzioni riportate nella scheda `eksctl`.

eksctl

1. Installare nuovamente il certificato. Sostituisci *`my-cluster`* con il nome del cluster.

   ```
   eksctl utils install-vpc-controllers -cluster my-cluster -approve
   ```

2. Verificare di ricevere l'output riportato di seguito.

   ```
   2021/05/28 05:24:59 [INFO] generate received request
   2021/05/28 05:24:59 [INFO] received CSR
   2021/05/28 05:24:59 [INFO] generating key: rsa-2048
   2021/05/28 05:24:59 [INFO] encoded CSR
   ```

3. Riavviare l'implementazione webhook.

   ```
   kubectl rollout restart deployment -n kube-system vpc-admission-webhook
   ```

4. Se il certificato che hai rinnovato è scaduto e hai Pods Windows bloccati nello stato `Container creating`, allora dovrai eliminare e implementare nuovamente quei Pods.

Windows

1. Ottenere lo script per generare un nuovo certificato.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.ps1;
   ```

2. Preparare il parametro per lo script.

   ```
   ./webhook-create-signed-cert.ps1 -ServiceName vpc-admission-webhook-svc -SecretName vpc-admission-webhook-certs -Namespace kube-system
   ```

3. Riavviare l'implementazione webhook.

```
kubectl rollout restart deployment -n kube-system vpc-admission-webhook-deployment
```

4. Se il certificato che hai rinnovato è scaduto e hai Pods Windows bloccati nello stato `Container creating`, allora dovrai eliminare e implementare nuovamente quei Pods.

Linux and macOS

Prerequisito

È necessario avere installato OpenSSL e `jq` sul computer.

1. Ottenere lo script per generare un nuovo certificato.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/manifests/region-code/vpc-admission-webhook/latest/webhook-create-signed-cert.sh
```

2. Modificare le autorizzazioni.

```
chmod +x webhook-create-signed-cert.sh
```

3. Eseguire lo script.

```
./webhook-create-signed-cert.sh
```

4. Riavviare il webhook.

```
kubectl rollout restart deployment -n kube-system vpc-admission-webhook-deployment
```

5. Se il certificato che hai rinnovato è scaduto e hai Pods Windows bloccati nello stato `Container creating`, allora dovrai eliminare e implementare nuovamente quei Pods.

Supporto di maggiore densità di Pod sui nodi di Windows

Su Amazon EKS a ciascun Pod viene assegnato un indirizzo `IPv4` dal tuo VPC. Per questo motivo, il numero di Pods che è possibile distribuire su un nodo è vincolato agli indirizzi IP disponibili, anche se ci sono risorse sufficienti per eseguire più Pods sul nodo. Poiché un nodo di Windows supporta una sola interfaccia di rete elastica, per impostazione predefinita, il numero massimo di indirizzi IP disponibili su un nodo di Windows è pari a:

```
Number of private IPv4 addresses for each interface on the node - 1
```

Un indirizzo IP viene utilizzato come indirizzo IP principale dell'interfaccia di rete, quindi non può essere assegnato a Pods.

Puoi abilitare una maggiore densità di Pod sui nodi di Windows abilitando la delega del prefisso IP. Questa funzionalità consente di assegnare un prefisso/28 `IPv4` all'interfaccia di rete principale, anziché assegnare indirizzi `IPv4` secondari. L'assegnazione di un prefisso IP aumenta il numero massimo di indirizzi `IPv4` disponibili sul nodo:

```
(Number of private IPv4 addresses assigned to the interface attached to the node - 1) *
 16
```

Con questo numero di indirizzi IP disponibili di gran lunga maggiore, gli indirizzi IP disponibili non dovrebbero limitare la capacità di dimensionare il numero di Pods sui tuoi nodi. Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2.

Requisiti dei cluster privati

Questo argomento descrive come distribuire un cluster Amazon EKS che viene distribuito su Cloud AWS, ma non dispone di accesso a Internet in uscita. Se hai un cluster locale attivo AWS OutpostsAvvio di nodi Amazon Linux autogestiti su un Outpost, consulta invece di questo argomento.

Se non si ha dimestichezza con la rete Amazon EKS, consultare Demistificazione delle reti cluster per i nodi (worker) Amazon EKS. Se il cluster non dispone di accesso a Internet in uscita, allora deve soddisfare i seguenti requisiti:

- Il tuo cluster deve estrarre immagini da un registro di container che si trova nel tuo VPC. Puoi creare un Amazon Elastic Container Registry nel VPC e copiare al suo interno le immagini dei container per i nodi da cui estrarre. Per ulteriori informazioni, consulta Copia di un'immagine di container da un repository a un altro.
- Il tuo cluster deve avere l'accesso privato all'endpoint abilitato. Ciò è obbligatorio per la registrazione dei nodi con l'endpoint del cluster. L'accesso pubblico all'endpoint è facoltativo. Per ulteriori informazioni, consulta Controllo accessi all'endpoint del cluster Amazon EKS.
- Prima di essere lanciati, i nodi Linux e Windows autogestiti devono includere i seguenti argomenti di bootstrap. Questi argomenti ignorano l'introspezione Amazon EKS e non richiedono l'accesso all'API Amazon EKS dal VPC.

1. Determina l'endpoint del cluster con il comando seguente. Sostituisci *my-cluster* con il nome del tuo cluster.

```
aws eks describe-cluster --name my-cluster --query cluster.endpoint --output text
```

   Di seguito viene riportato un output di esempio:

```
https://EXAMPLE108C897D9B2F1B21D5EXAMPLE.sk1.region-code.eks.amazonaws.com
```

2. Determina il valore dell'autorità di certificazione del cluster con il seguente comando. Sostituisci *my-cluster* con il nome del tuo cluster.

```
aws eks describe-cluster --name my-cluster --query cluster.certificateAuthority --output text
```

   L'output è una stringa molto lunga.

3. Sostituisci *cluster-endpoint* e *certificate-authority* nei comandi seguenti con i valori restituiti nell'output dei passaggi precedenti. Per ulteriori informazioni su come specificare gli argomenti di bootstrap quando si avviano nodi autogestiti, consulta Avvio di nodi Amazon Linux autogestiti e Avvio dei nodi Windows autogestiti.

- Per i nodi Linux:

```
--apiserver-endpoint cluster-endpoint --b64-cluster-ca certificate-authority
```

  Per maggiori informazioni, consulta lo script di bootstrap su GitHub.

- Per i nodi Windows:

> **Note**
>
> Se utilizzi il servizio personalizzato CIDR, devi specificarlo utilizzando il parametro `-ServiceCIDR`. In caso contrario, la risoluzione DNS per i Pods nel cluster fallirà.

```
-APIServerEndpoint cluster-endpoint -Base64ClusterCA certificate-authority
```

  Per ulteriori argomenti, consulta.Parametri di configurazione dello script di bootstrap.

- `aws-auth` `ConfigMap` del tuo cluster devono essere creati all'interno del tuo VPC. Per ulteriori informazioni sulla creazione e l'aggiunta di voci a `aws-auth` `ConfigMap`, digita **`eksctl create iamidentitymapping --help`** nel terminale. Se `ConfigMap` non esiste sul server, `eksctl` la creerà quando usi il comando per aggiungere una mappatura dell'identità.
- I Pods configurati con ruoli IAM per gli account di servizio acquisiscono le credenziali da una chiamata API AWS Security Token Service (AWS STS). Se non è disponibile un accesso a Internet in uscita, devi creare e utilizzare un endpoint AWS STS VPC nel tuo VPC. La maggior parte degli AWS `v1` SDK utilizza l' AWS STS endpoint globale di default (`sts.amazonaws.com`), che non utilizza l'endpoint VPC AWS STS . Per utilizzare l'endpoint AWS STS VPC, potrebbe essere necessario configurare l'SDK per utilizzare l'endpoint regionale AWS STS (). `sts.`*`region-code`*`.amazonaws.com` Per ulteriori informazioni, consulta Configurare l' AWS Security Token Service endpoint per un account di servizio.

- Le sottoreti VPC del cluster devono disporre di un endpoint di interfaccia VPC per qualsiasi Servizi AWS a cui Pods deve accedere. Per ulteriori informazioni, consulta Accesso a un servizio AWS utilizzando un endpoint VPC di interfaccia. Nella tabella seguente sono elencati alcuni servizi ed endpoint di uso comune. Per un elenco completo degli endpoint, consulta Servizi AWS che si integrano con AWS PrivateLink nella Guida di AWS PrivateLink.

Servizio	Endpoint
Amazon EC2	com.amazonaws.*codice-regione* .ec2
Amazon Elastic Container Registry (per estrarre immagini di container)	com.amazonaws.*codice-regione* .ecr.api, com.amazonaws.*codice-regione* .ecr.dkr e com.amazonaws.*codice-regione* .s3
Application Load Balancer e Network Load Balancer	com.amazonaws.*codice-regione* .elasticloadbalancing
AWS X-Ray	com.amazonaws.*codice-regione* .xray
CloudWatch Registri Amazon	com.amazonaws.*codice-regione* .logs

Servizio	Endpoint
AWS Security Token Service (richiesto quando si utilizzano i ruoli IAM per gli account di servizio)	com.amazonaws.*codice-regione* .sts

Considerazioni

- Qualsiasi nodo autogestito deve essere implementato in sottoreti con gli endpoint di interfaccia VPC richiesti. Se crei un gruppo di nodi gestiti, il gruppo di sicurezza dell'endpoint di interfaccia VPC deve consentire il CIDR per le sottoreti oppure dovrai aggiungere il gruppo di sicurezza del nodo creato al gruppo di sicurezza dell'endpoint di interfaccia VPC.
- Se Pods utilizzi volumi Amazon EFS, prima di distribuire il Driver CSI per Amazon EFS file kustomization.yaml del driver deve essere modificato per impostare le immagini dei container in modo che utilizzino le stesse del cluster Amazon EKS. Regione AWS
- Puoi utilizzare il AWS Load Balancer Controllerper distribuire AWS Application Load Balancers (ALB) e Network Load Balancers nel tuo cluster privato. Quando lo implementi, devi usare i flag della riga di comando per impostare`enable-shield`, `enable-waf` e `enable-wafv2` su fasle. Il rilevamento dei certificati con i nomi host degli oggetti in ingresso non è supportato. Questo perché il controller deve raggiungere un endpoint con interfaccia VPC AWS Certificate Manager, che non dispone di un endpoint con interfaccia VPC.

  Il controller supporta bilanciatori di carico di rete destinazioni IP, necessari per l'utilizzo con Fargate. Per ulteriori informazioni, consulta Bilanciamento del carico di applicazione su Amazon EKS e Creazione di un Network Load Balancer.
- Cluster Autoscaler è supportato. Quando implementi i Pods di Cluster Autoscaler, assicurati che la riga di comando includa `--aws-use-static-instance-list=true`. Per ulteriori informazioni, consulta Utilizzo dell'elenco di istanze statiche su GitHub. Il VPC del nodo di lavoro deve includere anche l'endpoint VPC e l'endpoint AWS STS VPC con scalabilità automatica.
- Alcuni prodotti software per container utilizzano chiamate API che accedono a per monitorare l'utilizzo. AWS Marketplace Metering Service I cluster privati non consentono queste chiamate, pertanto questi tipi di container non possono essere utilizzati nei cluster privati.

Versioni Kubernetes di Amazon EKS

Kubernetes è in rapida evoluzione, con nuovi aggiornamenti di progettazione, funzionalità e correzioni di bug. La community rilascia nuove versioni di Kubernetes secondarie (ad esempio, `1.30`) in media una volta ogni quattro mesi. Amazon EKS segue il ciclo di rilascio e deprecazione originario per le versioni secondarie. Man mano che nuove versioni di Kubernetes diventano disponibili in Amazon EKS, consigliamo di aggiornare tempestivamente i cluster in modo da usare la versione più recente disponibile.

Una versione secondaria è supportata come standard in Amazon EKS per i primi 14 mesi dopo il rilascio. Una volta superata la data di fine del supporto standard, la versione passa automaticamente al supporto esteso per i 12 mesi successivi. Il supporto esteso consente di mantenere una versione specifica di Kubernetespiù a lungo a un costo aggiuntivo per ora del cluster. Se non hai aggiornato il cluster prima della fine del periodo di supporto esteso, il cluster viene aggiornato automaticamente alla versione estesa più vecchia attualmente supportata.

Ti consigliamo di creare il cluster con l'ultima versione di Kubernetes disponibile supportata da Amazon EKS. Se l'applicazione richiede una versione specifica di Kubernetes, è possibile selezionare versioni precedenti. Puoi creare nuovi cluster Amazon EKS su qualsiasi versione offerta con supporto standard o esteso.

Versioni disponibili con supporto standard

Le seguenti versioni di Kubernetes sono attualmente disponibili nel supporto standard di Amazon EKS:

- `1.30`
- `1.29`
- `1.28`
- `1.27`
- `1.26`

Per le modifiche importanti di cui tenere conto per ogni versione del supporto standard, consulta Note di rilascio per versioni di supporto standard.

Versioni disponibili con supporto esteso

Le seguenti versioni di Kubernetes sono attualmente disponibili nel supporto esteso di Amazon EKS:

- `1.25`
- `1.24`
- `1.23`

Per conoscere le modifiche importanti di cui tenere conto per ogni versione del supporto esteso, consulta Note di rilascio per le versioni di supporto esteso.

Le seguenti Kubernetes versioni sono attualmente disponibili nel supporto esteso di Amazon EKS, con il requisito aggiuntivo che non è possibile creare nuovi cluster con queste versioni:

- `1.22`
- `1.21`

  Per informazioni su queste versioni, consulta Note di rilascio per le versioni 1.21 e 1.22

Calendario dei rilasci Kubernetes Amazon EKS

La tabella seguente mostra le date importanti di rilascio e supporto da considerare per ciascuna versione di Kubernetes.

> **Note**
>
> Le date con solo un mese e un anno sono approssimative e vengono aggiornate con una data esatta quando nota.

Versione Kubernetes	Rilascio a monte	Rilascio Amazon EKS	Data di fine del supporto standard	Data di fine del supporto esteso
`1.30`	17 aprile 2024	23 maggio 2024	23 luglio 2025	23 luglio 2026
`1.29`	13 dicembre 2023	23 gennaio 2024	23 marzo 2025	23 marzo 2026
`1.28`	15 agosto 2023	26 settembre 2023	26 novembre 2024	26 novembre 2025

Versione Kubernetes	Rilascio a monte	Rilascio Amazon EKS	Data di fine del supporto standard	Data di fine del supporto esteso
1.27	11 aprile 2023	24 maggio 2023	24 luglio 2024	24 luglio 2025
1.26	9 dicembre 2022	11 aprile 2023	11 giugno 2024	11 giugno 2025
1.25	23 agosto 2022	22 febbraio 2023	1 maggio 2024	1 maggio 2025
1.24	3 maggio 2022	15 novembre 2022	31 gennaio 2024	31 gennaio 2025
1.23	7 dicembre 2021	11 agosto 2022	11 ottobre 2023	11 ottobre 2024
1.22	4 agosto 2021	4 aprile 2022	4 giugno 2023	1 settembre 2024
1.21	8 aprile 2021	19 luglio 2021	16 febbraio 2023	15 luglio 2024

Domande frequenti sulle versioni di Amazon EKS

Quante versioni di Kubernetes sono disponibili nel supporto standard?

Amazon EKS, in linea con il supporto della community Kubernetes per le versioni di Kubernetes, si impegna a offrire il supporto standard per almeno quattro versioni di Kubernetes pronte per la produzione in qualunque momento. L'annuncio della data di fine supporto standard di una qualsiasi versione di Kubernetes secondaria sarà comunicato almeno 60 giorni prima. A causa del processo di qualifica e rilascio di Amazon EKS per le nuove versioni di Kubernetes, la data di fine supporto di una versione di Kubernetes in Amazon EKS avverrà in corrispondenza o dopo la data in cui il progetto Kubernetes smetterà di supportare la versione a monte.

Per quanto tempo Kubernetes riceve il supporto standard da parte di Amazon EKS?

Una versione di Kubernetes è supportata per 14 mesi dopo la prima disponibilità su Amazon EKS. Ciò è confermato anche se Kubernetes a monte non supporterà più una versione disponibile in Amazon EKS. Viene eseguito il backporting delle patch di sicurezza applicabili alle versioni di Kubernetes supportate in Amazon EKS.

Una volta terminato il supporto per una versione di Kubernetes su Amazon EKS ne riceverò notifica?

Sì. Se in alcuni cluster del tuo account è in esecuzione la versione prossima alla fine del supporto, Amazon EKS invia un avviso AWS Health Dashboard entro circa 12 mesi dal rilascio della Kubernetes versione su Amazon EKS. L'avviso include la data di fine supporto, che è successiva di almeno 60 giorni alla data dell'invio dell'avviso.

Quali funzionalità di Kubernetes sono supportate da Amazon EKS?

Amazon EKS supporta tutte le funzionalità di disponibilità generale (GA) dell'API di Kubernetes. A partire da Kubernetes versione `1.24`, per impostazione predefinita le nuove API beta non sono abilitate nei cluster. Tuttavia, per impostazione predefinita, le API beta precedentemente esistenti e le nuove versioni delle API beta esistenti continuano a essere abilitate. Alcune funzionalità alfa non sono supportate.

I gruppi di nodi gestiti da Amazon EKS vengono aggiornati in automatico insieme alla versione del piano di controllo del cluster?

No, un gruppo di nodi gestiti crea istanze Amazon EC2 nel tuo account. Queste istanze non vengono aggiornate in automatico quando l'utente o Amazon EKS aggiorna il piano di controllo. Per ulteriori informazioni, consulta Aggiornamento di un gruppo di nodi gestiti. Consigliamo di mantenere la stessa versione di Kubernetes sul piano di controllo e sui nodi.

I gruppi di nodi autogestiti vengono aggiornati in automatico insieme alla versione del piano di controllo del cluster?

No, un gruppo di nodi autogestito include le istanze di Amazon EC2 nel tuo account. Queste istanze non vengono aggiornate in automatico quando l'utente o Amazon EKS aggiorna la versione del piano di controllo per tuo conto. Nella console, un gruppo di nodi autogestito non riceve alcuna indicazione di aggiornamento. È possibile visualizzare la versione di `kubelet` installata su un nodo selezionando il nodo nella finestra di dialogo Nodi nella scheda Panoramica del cluster per determinare quali nodi devono essere aggiornati. È necessario aggiornare manualmente i nodi. Per ulteriori informazioni, consulta Aggiornamenti del nodo autogestito.

Il progetto Kubernetes verifica la compatibilità tra il piano di controllo e i nodi per un massimo di tre versioni secondarie. Ad esempio, i nodi `1.27` continuano a funzionare se orchestrati da un piano di controllo `1.30`. Tuttavia, non è consigliabile l'esecuzione di un cluster con nodi aggiornati in modo persistente a tre versioni secondarie precedenti rispetto al piano di controllo. Per ulteriori informazioni, consulta Policy per la versione Kubernetes e il supporto Skew della versione nella documentazione di Kubernetes. Si consiglia di mantenere la stessa versione di Kubernetes sul piano di controllo (control-plane) e sui nodi.

I Pods in esecuzione su Fargate vengono aggiornati in automatico con un aggiornamento automatico della versione del piano di controllo del cluster?

No. Consigliamo di eseguire i Pods Fargate come parte di un controller di replica, così come avviene in un'implementazione di Kubernetes. Quindi esegui un riavvio continuo di tutti i Pods Fargate. La nuova versione del Pod Fargate viene implementata con una versione di `kubelet` che corrisponde alla versione aggiornata del piano di controllo del cluster. Per ulteriori informazioni, consulta Implementazioni nella documentazione di Kubernetes.

> **Important**
>
> Se aggiorni il piano di controllo, dovrai comunque aggiornare personalmente i nodi Fargate. Per aggiornare i nodi Fargate, elimina il Pod Fargate rappresentato dal nodo e implementa nuovamente il Pod. Il nuovo Pod viene implementato con una versione `kubelet` che corrisponde alla versione del cluster.

Domande frequenti sul supporto esteso di Amazon EKS

La terminologia del supporto standard e del supporto esteso è nuova per me. Cosa significano questi termini?

Il supporto standard per una versione di Kubernetes in Amazon EKS inizia quando una Kubernetes versione viene rilasciata su Amazon EKS e termina 14 mesi dopo la data di rilascio. Il supporto esteso per una versione di Kubernetes inizierà immediatamente dopo la fine del supporto standard e terminerà trascorsi i 12 mesi successivi. Ad esempio, il supporto standard per la versione `1.23` in Amazon EKS terminerà l'11 ottobre 2023. Il supporto esteso per la versione è `1.23` iniziato il 12 ottobre 2023 e terminerà l'11 ottobre 2024.

Cosa devo fare per ottenere il supporto esteso per i cluster Amazon EKS?

Per ottenere un supporto esteso per i cluster Amazon EKS, non è necessario intraprendere alcuna azione. Il supporto standard inizia quando viene rilasciata una versione di Kubernetes su Amazon EKS e termina 14 mesi dopo la data di rilascio. Il supporto esteso per una versione di Kubernetes inizierà immediatamente dopo la fine del supporto standard e terminerà trascorsi i 12 mesi successivi. I cluster in esecuzione su una versione di Kubernetes successiva alla fine del supporto standard verranno automaticamente inclusi nel supporto esteso.

Per quali versioni Kubernetes posso ottenere il supporto esteso?

Il supporto esteso è disponibile per le versioni di Kubernetes `1.23` e successive. Puoi eseguire i cluster su qualsiasi versione per un massimo di 12 mesi dopo la fine del supporto standard per quella versione. Ciò significa che ogni versione sarà supportata per 26 mesi in Amazon EKS (14 mesi di supporto standard più 12 mesi di supporto esteso).

Cosa succede se non desidero usufruire del supporto esteso?

Se non desideri ricevere automaticamente il supporto esteso, puoi aggiornare il cluster a una versione di Kubernetes che includa il supporto standard di Amazon EKS. I cluster che non sono aggiornati a una versione di Kubernetes con supporto standard verranno inclusi automaticamente nel supporto esteso.

Cosa succederà alla fine dei 12 mesi di supporto esteso?

I cluster in esecuzione su una versione di Kubernetes che ha completato il ciclo di vita di 26 mesi (14 mesi di supporto standard più 12 mesi di supporto esteso) verranno aggiornati automaticamente alla versione successiva.

Allo scadere della data di fine di supporto, non è più possibile creare nuovi cluster Amazon EKS con la versione non supportata. I piani di controllo esistenti vengono aggiornati in automatico da Amazon EKS alla prima versione supportata attraverso un processo di implementazione graduale dopo la data di fine supporto. Dopo l'aggiornamento automatico del piano di controllo, è necessario aggiornare manualmente i componenti aggiuntivi del cluster e i nodi Amazon EC2. Per ulteriori informazioni, consulta Aggiornamento della versione di Kubernetes per il cluster Amazon EKS.

Allo scadere della data di fine supporto esteso, quando avverrà esattamente l'aggiornamento automatico del piano di controllo?

Amazon EKS non è in grado di fornire periodi di tempo specifici. Gli aggiornamenti automatici possono avvenire in qualsiasi momento dopo la data di fine supporto esteso. Non riceverai alcuna notifica prima dell'aggiornamento. Consigliamo di aggiornare in modo proattivo il piano di controllo senza fare affidamento sul processo di aggiornamento automatico di Amazon EKS. Per ulteriori informazioni, consulta Aggiornamento della versione di Kubernetes del cluster Amazon EKS.

Posso lasciare il mio piano di controllo su una versione di Kubernetes a tempo indeterminato?

No. La sicurezza del cloud AWS è la massima priorità. Dopo un certo periodo (di solito 1 anno), la community Kubernetes interrompe il rilascio di patch per vulnerabilità ed esposizioni (CVE) e sconsiglia l'invio di CVE per le versioni obsolete. Ciò significa che le vulnerabilità specifiche di una

versione precedente di Kubernetes potrebbero non essere segnalate, lasciando i cluster esposti senza preavviso e senza opzioni di correzione in caso di vulnerabilità. Pertanto, Amazon EKS non consente ai piani di controllo (control-plane) di rimanere a una versione non più coperta dal supporto esteso.

È previsto un costo aggiuntivo per ottenere il supporto esteso?

Sì, sono previsti costi aggiuntivi per i cluster Amazon EKS in esecuzione con supporto esteso. Per i dettagli sui prezzi, consulta il supporto esteso di Amazon EKS per i prezzi delle Kubernetes versioni sul AWS blog.

Cosa è incluso nel supporto esteso?

I cluster Amazon EKS in supporto esteso ricevono patch di sicurezza continue per il piano di controllo di Kubernetes. Inoltre, Amazon EKS rilascerà patch per Amazon VPC CNI, `kube-proxy` e componenti aggiuntivi per le versioni di supporto esteso su CoreDNS. Amazon EKS rilascerà anche patch per le AMI AWS ottimizzate Amazon EKS pubblicate per Amazon Linux e WindowsBottlerocket, oltre ai nodi Amazon EKS Fargate per tali versioni. Tutti i cluster di Extended Support continueranno ad avere accesso al supporto tecnico da AWS.

> **Note**
>
> L'Extended Support per le Windows AMI ottimizzate per Amazon EKS pubblicate da non AWS è disponibile per la Kubernetes versione`1.23`, ma è disponibile per le Kubernetes versioni `1.24` e successive.

Esistono limitazioni alle patch per i componenti di Kubernetes non inclusi nel supporto esteso?

Sebbene Extended Support copra tutti i componenti Kubernetes specifici di AWS, fornirà sempre supporto solo per AMI ottimizzate Amazon EKS AWS pubblicate per Amazon Linux e Windows. Bottlerocket Ciò significa che potresti avere componenti più recenti (come sistema operativo o kernel) sulla tua AMI ottimizzata per Amazon EKS durante l'utilizzo del supporto esteso. Ad esempio, quando Amazon Linux 2 raggiungerà la fine del suo ciclo di vita, nel 2025, le AMI Amazon Linux ottimizzate per Amazon EKS verranno create utilizzando un sistema operativo Amazon Linux più recente. Amazon EKS annuncerà e documenterà importanti discrepanze nel ciclo di vita del supporto, come questa, per ogni versione di Kubernetes.

Posso creare nuovi cluster utilizzando una versione con supporto esteso?

Sì, con l'esclusione di `1.22` e. `1.21` Ad esempio, puoi creare un `1.23` cluster, ma non un `1.22` cluster.

Note di rilascio per versioni di supporto standard

Questo argomento fornisce importanti modifiche di cui tenere conto per ogni versione Kubernetes del supporto standard. Durante l'aggiornamento, esamina attentamente le modifiche intervenute tra la vecchia e la nuova versione del cluster.

> **Note**
>
> Per `1.24` e cluster successivi, le AMI Amazon EKS pubblicate ufficialmente includono solo il runtime `containerd`. Le versioni di Kubernetes precedenti alla `1.24` utilizzano Docker come runtime predefinito. Queste versioni dispongono di un'opzione di flag di bootstrap che consente di testare i carichi di lavoro su qualsiasi cluster supportato con `containerd`. Per ulteriori informazioni, consulta Amazon EKS ha terminato il supporto per `Dockershim`.

Kubernetes1,30

Kubernetes `1.30` è ora disponibile in Amazon EKS. Per ulteriori informazioni su Kubernetes `1.30`, consulta l'annuncio del rilascio ufficiale.

> **Important**
>
> - A partire dalla versione Amazon EKS `1.30` o successiva, qualsiasi gruppo di nodi gestiti appena creato utilizzerà automaticamente Amazon Linux 2023 (AL2023) come sistema operativo del nodo. In precedenza, i nuovi gruppi di nodi utilizzavano per impostazione predefinita Amazon Linux 2 (AL2). Puoi continuare a utilizzare AL2 scegliendolo come tipo AMI durante la creazione di un nuovo gruppo di nodi.
>   - Per ulteriori informazioni su Amazon Linux, consulta Comparing AL2 e AL2023 nella Amazon Linux User Guide.
>   - Per ulteriori informazioni sulla specificazione del sistema operativo per un gruppo di nodi gestiti, consulta. Creazione di un gruppo di nodi gestiti

- Con Amazon EKS`1.30`, l'`topology.k8s.aws/zone-id`etichetta viene aggiunta ai nodi di lavoro. Puoi utilizzare gli ID delle zone di disponibilità (ID AZ) per determinare la posizione delle risorse in un account rispetto alle risorse di un altro account. Per ulteriori informazioni, consulta gli ID delle zone di disponibilità per AWS le tue risorse nella Guida per l'AWS RAM utente.
- A partire da`1.30`, Amazon EKS non include più l'`default`annotazione sulla `gp2` `StorageClass` risorsa applicata ai cluster appena creati. Ciò non ha alcun impatto se si fa riferimento a questa classe di storage per nome. È necessario agire se si contava sulla presenza di un valore predefinito `StorageClass` nel cluster. È necessario fare riferimento a `StorageClass` con il nome`gp2`. In alternativa, puoi distribuire la classe di storage predefinita consigliata da Amazon EBS impostando il `defaultStorageClass.enabled` parametro su true durante l'installazione `v1.31.0` o successivamente di. `aws-ebs-csi-driver add-on`
- La policy IAM minima richiesta per il ruolo IAM del cluster Amazon EKS è cambiata. L'azione `ec2:DescribeAvailabilityZones` è obbligatoria. Per ulteriori informazioni, consulta Ruolo IAM del cluster Amazon EKS.

Per il changelog completo di Kubernetes `1.30`, consulta https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.30.md.

Kubernetes1.29

Kubernetes `1.29` è ora disponibile in Amazon EKS. Per ulteriori informazioni su Kubernetes `1.29`, consulta l'annuncio del rilascio ufficiale.

> **Important**
>
> - La versione `flowcontrol.apiserver.k8s.io/v1beta2` API obsoleta di `FlowSchema` e non è più disponibile `PriorityLevelConfiguration` in. Kubernetes `v1.29` Se disponi di manifesti o di un software client che utilizza il gruppo di API beta obsoleto, dovresti modificarli prima di eseguire l'aggiornamento a. `v1.29`

- Il `.status.kubeProxyVersion` campo per gli oggetti del nodo è ora obsoleto e il Kubernetes progetto propone di rimuovere quel campo in una versione futura. Il campo obsoleto non è preciso e storicamente è stato gestito da, che in realtà non conosce la `kubelet` versione e nemmeno se sia in esecuzione. `kube-proxy` `kube-proxy` Se hai utilizzato questo campo nel software client, smettila: le informazioni non sono affidabili e il campo è ora obsoleto.

- Kubernetes`1.29`Per ridurre la potenziale superficie di attacco, la `LegacyServiceAccountTokenCleanUp` funzionalità etichetta i token basati su segreti generati automaticamente come non validi se non sono stati utilizzati per un lungo periodo (1 anno per impostazione predefinita) e li rimuove automaticamente se non si tenta di utilizzarli per un lungo periodo dopo essere stati contrassegnati come non validi (1 anno aggiuntivo per impostazione predefinita). Per identificare tali token, puoi eseguire:

```
kubectl get cm kube-apiserver-legacy-service-account-token-tracking -nkube-system
```

Per il changelog completo di Kubernetes `1.29`, consulta https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.29.md#changelog-since-v1280.

Kubernetes1.28

Kubernetes `1.28` è ora disponibile in Amazon EKS. Per ulteriori informazioni su Kubernetes `1.28`, consulta l'annuncio del rilascio ufficiale.

- Kubernetes `v1.28` ha ampliato l'inclinazione supportata tra i componenti del nodo principale e del piano di controllo (control-plane) con una versione secondaria, da `n-2` a `n-3`, in modo che i componenti del nodo (`kubelet` e `kube-proxy`) per la versione secondaria supportata più vecchia possano funzionare con i componenti del piano di controllo (control-plan) (`kube-apiserver`, `kube-scheduler`, `kube-controller-manager`, `cloud-controller-manager`) per la versione minore supportata più recente.
- I parametri `force_delete_pods_total` e `force_delete_pod_errors_total` nel `Pod GC Controller` sono stati migliorati per tenere conto dell'eliminazione forzata di tutti i pod. Viene aggiunto un motivo alla metrica per indicare se il pod viene eliminato forzatamente perché è terminato, è orfano, presenta la contaminazione o è terminato e non è pianificato. out-of-service
- Il controller `PersistentVolume (PV)` è stato modificato per assegnare automaticamente un valore predefinito `StorageClass` a qualsiasi `PersistentVolumeClaim` non associato con lo `storageClassName` non impostato. Inoltre, il meccanismo di convalida dell'ammissione `PersistentVolumeClaim` all'interno del server API è stato modificato per consentire la modifica dei valori da uno stato non impostato a un nome `StorageClass` effettivo.

Per il changelog completo di Kubernetes `1.28`, consulta https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.28.md#changelog-since-v1270.

Kubernetes1.27

Kubernetes `1.27` è ora disponibile in Amazon EKS. Per ulteriori informazioni su Kubernetes `1.27`, consulta l'annuncio del rilascio ufficiale.

Important

- Il supporto per le annotazioni `seccomp` alfa e le annotazioni `seccomp.security.alpha.kubernetes.io/pod` e `container.seccomp.security.alpha.kubernetes.io` è stato rimosso. Le annotazioni `seccomp` alfa sono state dichiarate obsolete in `1.19`, e una volta rimosse in `1.27`, i campi `seccomp` non verranno più compilati automaticamente per i `Pods` con le annotazioni `seccomp`. Utilizza invece il campo `securityContext.seccompProfile` per i `Pods` o i container per configurare i profili `seccomp`. Per verificare se stai utilizzando le annotazioni `seccomp` alfa obsolete nel tuo cluster, esegui il comando seguente:

  ```
  kubectl get pods --all-namespaces -o json | grep
   -E 'seccomp.security.alpha.kubernetes.io/pod|
  container.seccomp.security.alpha.kubernetes.io'
  ```

- L'argomento della linea di comando `--container-runtime` per `kubelet` è stato rimosso. Il runtime del contenitore predefinito per Amazon EKS esiste da `containerd` allora`1.24`, il che elimina la necessità di specificare il runtime del contenitore. Dalla versione `1.27` in avanti, Amazon EKS ignorerà l'argomento `--container-runtime` passato a qualsiasi script di bootstrap. È importante non trasmettere questo argomento a `--kubelet-extra-args` per evitare che si verifichino errori durante il processo di avvio del nodo. È necessario rimuovere l'argomento `--container-runtime` da tutti i flussi di lavoro di creazione dei nodi e dagli script di creazione.

- Il `kubelet` in Kubernetes `1.27` ha aumentato `kubeAPIQPS` predefinito a `50` e `kubeAPIBurst` a `100`. Questi miglioramenti consentono al `kubelet` di gestire un volume maggiore di query API, migliorando i tempi di risposta e le prestazioni. Quando le richieste ai `Pods` aumentano, a causa dei requisiti di dimensionamento, le impostazioni predefinite riviste garantiscono che il `kubelet` possa gestire in modo efficiente l'aumento del carico di lavoro. Di conseguenza, gli avvii dei Pod sono più rapidi e le operazioni sui cluster sono più efficaci.

- È possibile utilizzare una topologia del Pod più dettagliata per distribuire le policy, come `minDomain`. Questo parametro consente di specificare il numero minimo di domini su cui i `Pods` dovrebbero essere distribuiti. `nodeAffinityPolicy` e `nodeTaintPolicy` forniscono un ulteriore livello di granularità nella gestione della distribuzione dei Pod. Ciò avviene in conformità alle affinità dei nodi, ai taint e al campo `matchLabelKeys` indicati nel `topologySpreadConstraints` della specifica del `Pod's`. Ciò consente di selezionare i `Pods` per i calcoli di distribuzione dopo un aggiornamento continuo.
- Kubernetes`1.27` ha promosso a beta un nuovo meccanismo di policy per `StatefulSets` che controlli la durata di `PersistentVolumeClaims` (PVCs). La nuova policy di conservazione di `PVC` consente di specificare se i `PVCs` generati dal modello di specifica `StatefulSet` verranno eliminati o conservati automaticamente quando `StatefulSet` viene eliminato o le repliche contenute in `StatefulSet` vengono ridimensionate.
- L'opzione `goaway-chance` nel server API Kubernetes aiuta a evitare che le connessioni client `HTTP/2` rimangano bloccate su una singola istanza del server API, chiudendo casualmente una connessione. Quando la connessione viene chiusa, il client tenterà di riconnettersi e probabilmente giungerà su un server API diverso a seguito del sistema di bilanciamento del carico. Amazon EKS versione `1.27` ha abilitato il flag di `goaway-chance`. Se il carico di lavoro in esecuzione sul cluster Amazon EKS utilizza un client non compatibile con `HTTP GOAWAY`, è preferibile aggiornare il client in modo che gestisca `GOAWAY` ristabilendo la connessione quando questa si arresta.

Per il changelog completo di Kubernetes `1.27`, consulta https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.27.md#changelog-since-v1260.

Kubernetes1.26

Kubernetes `1.26` è ora disponibile in Amazon EKS. Per ulteriori informazioni su Kubernetes `1.26`, consulta l'annuncio del rilascio ufficiale.

> **Important**
>
> Kubernetes `1.26` non supporta più CRI `v1alpha2`. Ciò comporta che il `kubelet` non registri più il nodo se il runtime del container non supporta CRI `v1`. Ciò significa anche che Kubernetes `1.26` non supporta la versione secondaria containerd `1.5` e precedenti. Se utilizzi containerd, è necessario eseguire l'aggiornamento alla versione containerd `1.6.0` o a una versione successiva prima di aggiornare qualsiasi nodo a Kubernetes `1.26`. È inoltre necessario aggiornare qualsiasi altro runtime di container che supporti solo il `v1alpha2`.

Per ulteriori informazioni, rivolgiti al fornitore del runtime del container. Per impostazione predefinita, le AMI Amazon Linux e Bottlerocket includono la versione containerd `1.6.6`.

- Prima di eseguire l'aggiornamento a Kubernetes `1.26`, esegui l'aggiornamento di Amazon VPC CNI plugin for Kubernetes alla versione `1.12` o a una versione successiva. Se non esegui l'upgrade aAmazon VPC CNI plugin for Kubernetesversione`1.12`o più tardi, ilAmazon VPC CNI plugin for Kubernetessi bloccherà. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.
- L'opzione `goaway-chance` nel server API Kubernetes aiuta a evitare che le connessioni client `HTTP/2` rimangano bloccate su una singola istanza del server API, chiudendo casualmente una connessione. Quando la connessione viene chiusa, il client tenterà di riconnettersi e probabilmente giungerà su un server API diverso a seguito del sistema di bilanciamento del carico. Amazon EKS versione `1.26` ha abilitato il flag di `goaway-chance`. Se il carico di lavoro in esecuzione sul cluster Amazon EKS utilizza un client non compatibile con `HTTP GOAWAY`, è preferibile aggiornare il client in modo che gestisca `GOAWAY` ristabilendo la connessione quando questa si arresta.

Per il changelog completo di Kubernetes `1.26`, consulta https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.26.md#changelog-since-v1250.

Note di rilascio per le versioni di supporto esteso

Questo argomento fornisce importanti modifiche di cui tenere conto per ogni versione di Kubernetes del supporto esteso. Durante l'aggiornamento, esamina attentamente le modifiche apportate tra la vecchia e la nuova versione del cluster.

Kubernetes1,25

Kubernetes `1.25` è ora disponibile in Amazon EKS. Per ulteriori informazioni su Kubernetes `1.25`, consulta l'annuncio del rilascio ufficiale.

Important

- A partire da Kubernetes versione `1.25`, non sarai più in grado di utilizzare le istanze `P2` di Amazon EC2 con le AMI Amazon Linux accelerate ottimizzate e preconfigurate per Amazon EKS. Queste AMI per versioni di Kubernetes `1.25` o successive supporteranno driver di serie `NVIDIA 525` o successive, che non sono compatibili con le istanze `P2`.

Tuttavia, i driver della serie `NVIDIA 525` o versioni successive sono compatibili con le istanze `P3`, `P4` e `P5`, quindi puoi utilizzare quelle istanze con le AMI per Kubernetes versione `1.25`. Prima che i cluster Amazon EKS vengano aggiornati alla versione `1.25`, migrano qualsiasi istanza `P2` su istanze `P3`, `P4`, e `P5`. È inoltre necessario aggiornare in modo proattivo le applicazioni per funzionare con la serie `NVIDIA 525` o successive. Prevediamo di effettuare il backport dei driver della `NVIDIA 525` serie più recente o successiva alle Kubernetes versioni `1.23` e alla `1.24` fine di gennaio 2024.

- `PodSecurityPolicy` (PSP) viene rimossa in Kubernetes `1.25`. Le PSPs sono sostituite da Pod Security Admission (PSA) e Pod Security Standards (PSS). PSA è un controller di ammissione integrato che implementa i controlli di sicurezza descritti in PSS. PSA e PSS sono passati allo stato stabile in Kubernetes `1.25` e sono abilitati in Amazon EKS per impostazione predefinita. Se disponi PSPs di un cluster, assicurati di migrare dalla versione integrata Kubernetes PSS o PSP a una policy-as-code soluzione prima di aggiornare il cluster alla versione precedente. `1.25` Se non esegui la migrazione da PSP, potresti riscontrare interruzioni dei carichi di lavoro. Per ulteriori informazioni, consulta Domande frequenti sulla policy di sicurezza pod (PSP).
- Kubernetesversione`1.25`contiene modifiche che alterano il comportamento di una funzionalità esistente nota come API Priority and Fairness (APF). L'APF serve a proteggere il server API da un potenziale sovraccarico durante i periodi di elevati volumi di richieste. Lo fa imponendo restrizioni al numero di richieste simultanee che possono essere elaborate in un dato momento. Ciò si ottiene mediante l'applicazione di livelli di priorità e limiti distinti alle richieste provenienti da vari carichi di lavoro o utenti. Questo approccio garantisce che le applicazioni critiche o le richieste ad alta priorità ricevano un trattamento preferenziale, evitando allo stesso tempo che le richieste con priorità inferiore sovraccarichino il server API. Per ulteriori informazioni, vederePriorità ed equità delle APInelKubernetesdocumentazione oPriorità ed equità delle APInella Guida alle migliori pratiche di EKS.

  Questi aggiornamenti sono stati introdotti inO #10352eO #118601. In precedenza, APF trattava tutti i tipi di richieste in modo uniforme, con ogni richiesta che consumava una singola unità del limite di richieste simultanee. La modifica del comportamento APF assegna unità di concorrenza più elevate a`LIST`richieste dovute all'onere eccezionalmente pesante imposto al server API da tali richieste. Il server API stima il numero di oggetti che verranno restituiti da un`LIST`richiesta. Assegna un'unità di concorrenza proporzionale al numero di oggetti restituiti.

Dopo l'aggiornamento alla versione Amazon EKS`1.25`o superiore, questo comportamento aggiornato potrebbe causare carichi di lavoro pesanti`LIST`richieste (che in precedenza funzionavano senza problemi) di riscontrare una limitazione della velocità. Ciò verrebbe indicato da un codice di risposta HTTP 429. Per evitare potenziali interruzioni del carico di lavoro dovute a`LIST`poiché le richieste sono a tariffa limitata, ti consigliamo vivamente di ristrutturare i tuoi carichi di lavoro per ridurne la frequenza. In alternativa, puoi risolvere questo problema modificando le impostazioni APF per allocare più capacità per le richieste essenziali riducendo al contempo la capacità allocata a quelle non essenziali. Per ulteriori informazioni su queste tecniche di mitigazione, vederePrevenzione delle richieste persenella Guida alle migliori pratiche di EKS.

- Amazon EKS `1.25` include miglioramenti all'autenticazione dei cluster che contengono librerie aggiornate YAML. Se un valore YAML in `aws-auth` `ConfigMap` disponibile nello spazio dei nomi `kube-system` inizia con una macro, dove il primo carattere è una parentesi graffa, è necessario aggiungere le virgolette (`" "`) prima e dopo le parentesi graffe (`{ }`). Ciò è necessario per garantire che `aws-iam-authenticator` versione `v0.6.3` analizzi accuratamente `aws-auth` `ConfigMap` in Amazon EKS `1.25`.
- La versione beta dell'API (`discovery.k8s.io/v1beta1`) di `EndpointSlice` è stata dichiarata obsoleta in Kubernetes `1.21` e non è più disponibile in Kubernetes `1.25`. Questa API è stata aggiornata a `discovery.k8s.io/v1`. Per ulteriori informazioni, consulta `EndpointSlice` nella documentazione Kubernetes. AWS Load Balancer Controller `v2.4.6` e versioni precedenti utilizzavano l'endpoint `v1beta1` per comunicare con `EndpointSlices`. Se stai utilizzando la configurazione `EndpointSlices` per AWS Load Balancer Controller, devi eseguire l'aggiornamento a AWS Load Balancer Controller `v2.4.7` prima di aggiornare il cluster Amazon EKS a `1.25`. Se si esegue l'aggiornamento a `1.25` mentre si utilizza la configurazione `EndpointSlices` per AWS Load Balancer Controller, il controller si arresterà e causerà interruzioni dei carichi di lavoro. Per aggiornare il controller, consulta Che cosa è la AWS Load Balancer Controller?.

- `SeccompDefault` viene promosso alla versione beta in Kubernetes `1.25`. Impostando il flag `--seccomp-default` durante la configurazione di `kubelet`, il runtime del container utilizza il suo profilo `RuntimeDefault` `seccomp`, anziché la modalità unconfined (`seccomp disabled`). I profili predefiniti forniscono una serie completa di impostazioni predefinite di sicurezza, preservando al contempo la funzionalità del carico di lavoro. Sebbene questo flag sia disponibile, Amazon EKS non lo abilita per impostazione predefinita, quindi il comportamento di Amazon EKS

rimane effettivamente invariato. Se lo desideri, puoi iniziare ad abilitarlo sui nodi. Per maggiori dettagli, consulta il tutorial Limitare i syscall di un container con seccomp nella documentazione Kubernetes.

- Il supporto per CRI (Container Runtime Interface) per Docker (detto anche `Dockershim`) è stato rimosso da Kubernetes `1.24` e versioni successive. L'unico runtime del container nelle AMIs Amazon EKS ufficiale per Kubernetes `1.24` e versioni successive è `containerd`. Prima eseguire l'upgrade ad Amazon EKS `1.24` o versioni successive, devi rimuovere qualsiasi riferimento ai flag degli script di bootstrap non più supportati. Per ulteriori informazioni, consulta Amazon EKS ha terminato il supporto per `Dockershim`.
- Il supporto per le query con caratteri jolly è stato dichiarato obsoleto in CoreDNS `1.8.7` e rimosso in CoreDNS `1.9`. Questo è stato fatto come misura di sicurezza. Le query con caratteri jolly non funzionano più e restituiscono `NXDOMAIN` invece di un indirizzo IP.
- L'opzione `goaway-chance` nel server API Kubernetes aiuta a evitare che le connessioni client `HTTP/2` rimangano bloccate su una singola istanza del server API, chiudendo casualmente una connessione. Quando la connessione viene chiusa, il client tenterà di riconnettersi e probabilmente giungerà su un server API diverso a seguito del sistema di bilanciamento del carico. Amazon EKS versione `1.25` ha abilitato il flag di `goaway-chance`. Se il carico di lavoro in esecuzione sul cluster Amazon EKS utilizza un client non compatibile con `HTTP GOAWAY`, è preferibile aggiornare il client in modo che gestisca `GOAWAY` ristabilendo la connessione quando questa si arresta.

Per il changelog completo di Kubernetes `1.25`, consulta https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.25.md#changelog-since-v1240.

Kubernetes1.24

Kubernetes `1.24` è ora disponibile in Amazon EKS. Per ulteriori informazioni su Kubernetes `1.24`, consulta l'annuncio del rilascio ufficiale.

> **Important**
>
> - A partire da Kubernetes `1.24`, per impostazione predefinita le nuove API beta non sono abilitate nei cluster. Per impostazione predefinita, le API beta esistenti e le nuove versioni delle API beta esistenti continuano a essere abilitate. Amazon EKS segue lo stesso funzionamento di Kubernetes `1.24` upstream. I feature gate che controllano le nuove funzionalità per le operazioni API nuove ed esistenti sono abilitati per impostazione

predefinita. Questo è in linea con Kubernetes upstream. Per ulteriori informazioni, consulta KEP-3136: le API beta sono disattivate per impostazione predefinita. GitHub

- Il supporto per CRI (Container Runtime Interface) per Docker (detto anche Dockershim) è rimosso da Kubernetes `1.24`. Le AMI ufficiali di Amazon EKS includono solo il runtime containerd. Prima di passare ad Amazon EKS `1.24` o versioni successive, devi rimuovere qualsiasi riferimento ai flag degli script di bootstrap non più supportati. Inoltre è necessario assicurarsi che l'inoltro di IP sia abilitato per i nodi worker. Per ulteriori informazioni, consulta Amazon EKS ha terminato il supporto per `Dockershim`.
- Se hai già configurato Fluentd per Container Insights, devi eseguire la migrazione di Fluentd a Fluent Bit prima di aggiornare il cluster. I parser Fluentd sono configurati per analizzare solo i messaggi di log in formato JSON. A differenza di `dockerd`, il runtime del container `containerd` contiene messaggi di log che non sono in formato JSON. Se non esegui la migrazione a Fluent Bit, alcuni dei parser Fluentd's configurati genereranno un'enorme quantità di errori all'interno del container Fluentd. Per ulteriori informazioni sulla migrazione, consulta Configurare Fluent Bit come invio di log DaemonSet a Logs. CloudWatch
- In Kubernetes `1.23` e versioni precedenti, i certificati di servizio `kubelet` con IP non verificabili e nomi alternativi del soggetto (SAN) DNS vengono emessi automaticamente con SAN non verificabili. Questi SAN non verificabili sono omessi dal certificato fornito. Nei cluster `1.24` e versioni successive, i certificati di servizio `kubelet` non vengono emessi se non è possibile verificare un SAN. Ciò impedisce il funzionamento dei comandi dei log di `kubectl` e `kubectl`. Per ulteriori informazioni, consulta Considerazioni sulla firma dei certificati prima di aggiornare il cluster alla versione 1.24 di Kubernetes.
- Quando aggiorni un cluster Amazon EKS `1.23` che utilizza Fluent Bit, è necessario assicurarsi che sia in esecuzione una versione `k8s/1.3.12` o precedente. Puoi farlo riapplicando l'ultimo file YAML Fluent Bit applicabile da GitHub. Per ulteriori informazioni, consulta Configurazione Fluent Bit nella Amazon CloudWatch User Guide.

- Puoi utilizzare Topology Aware Hints per indicare la tua preferenza per mantenere il traffico in una zona quando i nodi worker del cluster vengono implementati in più zone di disponibilità. L'instradamento del traffico all'interno di una zona può contribuire a ridurre i costi e migliorare le prestazioni della rete. Per impostazione predefinita, i Topology Aware Hints sono abilitati in Amazon EKS `1.24`. Per ulteriori informazioni, consulta Topology Aware Hints nella documentazione di Kubernetes.

- La rimozione di `PodSecurityPolicy` (PSP) è prevista in Kubernetes `1.25`. I PSPs verranno sostituiti con PSA (Pod Security Admission). PSA è un controller di ammissione integrato che utilizza i controlli di sicurezza descritti in PSS (Pod Security Standards). PSA e PSS sono entrambe funzionalità beta e sono abilitate in Amazon EKS per impostazione predefinita. Per gestire la rimozione di PSP nella versione `1.25`, è preferibile implementare PSS in Amazon EKS. Per ulteriori informazioni, consulta Implementing Pod Security Standards in Amazon EKS (Implementazione degli standard di sicurezza Pod in Amazon EKS) nel blog AWS .
- `client.authentication.k8s.io/v1alpha1` ExecCredential Viene rimosso in Kubernetes`1.24`. L' ExecCredential API era generalmente disponibile in Kubernetes`1.22`. Se utilizzi un plugin di credenziali client-go che si basa sull'API `v1alpha1`, contatta il distributore del plugin per la modalità con cui eseguire la migrazione all'API `v1`.
- Per Kubernetes `1.24`, abbiamo contribuito con una funzionalità al progetto upstream Cluster Autoscaler che semplifica il dimensionamento dei gruppi di nodi gestiti da Amazon EKS da/verso zero nodi. In precedenza, per consentire a Cluster Autoscaler di comprendere le risorse, le etichette e i taint di un gruppo di nodi gestiti che era scalato a zero nodi, era necessario applicare tag al gruppo Dimensionamento automatico Amazon EC2 sottostante con i dettagli dei nodi di cui era responsabile. Attualmente, quando non esistono nodi in esecuzione nel gruppo di nodi gestiti, Cluster Autoscaler richiama l'operazione dell' API `DescribeNodegroup` di Amazon EKS. Questa operazione dell'API fornisce le informazioni richieste da Cluster Autoscaler sulle risorse, sulle etichette e sui taint del gruppo di nodi gestiti. Questa funzionalità richiede l'aggiunta dell'autorizzazione `eks:DescribeNodegroup` alla policy IAM dell'account di servizio Cluster Autoscaler. Quando il valore di un tag Cluster Autoscaler sul gruppo con scalabilità automatica che alimenta un gruppo di nodi gestiti da Amazon EKS è in conflitto con il gruppo di nodi stesso, Cluster Autoscaler preferisce il valore del tag del gruppo con scalabilità automatica. Ciò ti consente di sovrascrivere i valori in base alla necessità. Per ulteriori informazioni, consulta Scalabilità automatica.
- Se intendi utilizzare i nostri Inferentia tipi di Trainium istanze con Amazon EKS`1.24`, devi eseguire l'aggiornamento alla versione 1.9.3.0 o successiva del plug-in del AWS Neuron dispositivo. Per ulteriori informazioni, consulta la versione di Neuron K8 [1.9.3.0] nella documentazione. AWS Neuron
- Per impostazione predefinita, `Containerd` dispone di `IPv6` abilitato per i Pods. Applica le impostazioni del kernel dei nodi agli spazi dei nomi di rete del Pod. Per questo motivo, i container in un Pod si associano agli indirizzi di loopback sia `IPv4` (`127.0.0.1`) sia `IPv6` (`::1`). `IPv6` è il protocollo di comunicazione predefinito. Prima di aggiornare il cluster alla versione `1.24`, ti consigliamo di testare i Pods multi-container. Modifica le app in modo che possano associarsi a tutti gli indirizzi IP sulle interfacce di loopback. La maggior parte delle librerie abilita l'associazione

IPv6, che è compatibile con IPv4. Quando non è possibile modificare il codice dell'applicazione, sono disponibili due opzioni:

- Eseguire un container init e impostare disable ipv6 su true (sysctl -w net.ipv6.conf.all.disable_ipv6=1).
- Configurare un webhook di ammissione variabile per inserire un container init insieme ai Pods delle applicazioni.

Se devi bloccare IPv6 per tutti i Pods su tutti i nodi, potrebbe essere necessario disabilitare IPv6 nelle tue istanze.

- L'opzione goaway-chance nel server API Kubernetes aiuta a evitare che le connessioni client HTTP/2 rimangano bloccate su una singola istanza del server API, chiudendo casualmente una connessione. Quando la connessione viene chiusa, il client tenterà di riconnettersi e probabilmente giungerà su un server API diverso a seguito del sistema di bilanciamento del carico. Amazon EKS versione 1.24 ha abilitato il flag di goaway-chance. Se il carico di lavoro in esecuzione sul cluster Amazon EKS utilizza un client non compatibile con HTTP GOAWAY, è preferibile aggiornare il client in modo che gestisca GOAWAY ristabilendo la connessione quando questa si arresta.

Per il changelog completo di Kubernetes 1.24, consulta https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.24.md#changelog-since-v1230.

Kubernetes1.23

Kubernetes 1.23 è ora disponibile in Amazon EKS. Per ulteriori informazioni su Kubernetes 1.23, consulta l'annuncio del rilascio ufficiale.

> **Important**
>
> - È abilitata la funzionalità di migrazione del volume da Kubernetes nella struttura a CSI (Container Storage Interface). Questa funzionalità consente la sostituzione dei plug-in di archiviazione predefiniti di Kubernetes per Amazon EBS con un driver CSI di Amazon EBS corrispondente. Per ulteriori informazioni, consulta Funzionalità Kubernetes 1.17: la migrazione dai volumi predefiniti Kubernetes a CSI passa alla versione beta sul blog Kubernetes.
>
>   La funzione traduce le API predefinite in API CSI equivalenti e delega le operazioni a un driver CSI sostitutivo. Con questa funzione, se utilizzi oggetti StorageClass, PersistentVolume e PersistentVolumeClaim esistenti che appartengono a questi

carichi di lavoro, probabilmente non ci saranno cambiamenti evidenti. La funzione consente a Kubernetes di delegare tutte le operazioni di gestione dello spazio di archiviazione dal plug-in predefinito al driver CSI. Se utilizzi volumi Amazon EBS in un cluster esistente, installa il driver CSI Amazon EBS nel tuo cluster prima di aggiornare il cluster alla versione `1.23`. Se non si installa il driver prima di aggiornare il cluster, potrebbero verificarsi interruzioni dei carichi di lavoro. Se prevedi di implementare carichi di lavoro che utilizzano volumi Amazon EBS in un nuovo cluster `1.23`, installa il driver CSI Amazon EBS nel cluster prima di implementare i carichi di lavoro del cluster. Per ricevere istruzioni sull'installazione del driver CSI di Amazon EBS nel cluster, consulta Driver CSI per Amazon EBS. Per le domande frequenti sulla funzionalità di migrazione, consulta Domande frequenti sulla migrazione CSI di Amazon EBS.

- L'Extended Support per le Windows AMI ottimizzate per Amazon EKS pubblicate da non AWS è disponibile per la Kubernetes versione`1.23`, ma è disponibile per le Kubernetes versioni `1.24` e successive.

- Kubernetes ha smesso di supportare `dockershim` nella versione `1.20` e ha rimosso `dockershim` nella versione `1.24`. Per ulteriori informazioni, consulta l'articolo KubernetesKubernetes sta abbandonando Dockershim: impegni e fasi successive nel blog Kubernetes. Amazon EKS interromperà il supporto per `dockershim` a partire da Amazon EKS versione `1.24`. A partire dalla versione `1.24` di Amazon EKS, le AMI ufficiali di Amazon EKS avranno `containerd` come unico runtime.

  Anche se Amazon EKS versione `1.23` continua a supportare `dockershim`, è preferibile iniziare subito a testare le applicazioni per identificare e rimuovere eventuali dipendenze da Docker. In questo modo, sarai preparato ad aggiornare il tuo cluster alla versione `1.24`. Per ulteriori informazioni sulla rimozione di `dockershim`, consulta Amazon EKS ha terminato il supporto per `Dockershim`.
- Kubernetes ha promosso le reti dual-stack `IPv4`/`IPv6` per i servizi Pods e i nodi alla disponibilità generale. Tuttavia, Amazon EKS e il Amazon VPC CNI plugin for Kubernetes al momento non supportano la rete dual-stack. I tuoi cluster possono assegnare indirizzi `IPv4` o `IPv6` a Pods e servizi, ma non possono assegnare entrambi i tipi di indirizzo.
- Kubernetes ha portato la funzione PSA (Pod Security Admission) alla versione beta. Questa funzionalità è abilitata per impostazione predefinita. Per ulteriori informazioni, consulta Ammissione di sicurezza dei pod nella documentazione di Kubernetes. PSA sostituisce il controller di

ammissione Pod Security Policy (PSP). Il controller di ammissione PSP non è supportato e la sua rimozione è pianificata per Kubernetes versione `1.25`.

Il controller di ammissione PSP applica gli standard di sicurezza Pod su Pods in uno spazio dei nomi basato su specifiche etichette dello spazio dei nomi che impostano il livello di applicazione. Per ulteriori informazioni, consulta Pod Security Standards (PSS) e Pod Security Admission (PSA) nella guida alle best practice per Amazon EKS.

- L'immagine `kube-proxy` implementata con i cluster è ora l'immagine di base minima gestita da Amazon EKS Distro (EKS-D). L'immagine contiene pacchetti minimi, senza shell (interprete di comandi) né gestori di pacchetti.
- Kubernetes ha promosso i container temporanei alla versione beta. I container temporanei sono container che vengono eseguiti nello stesso spazio dei nomi di un Pod esistente. Puoi utilizzarli per osservare lo stato di Pods e container per la risoluzione dei problemi e il debug. Ciò è utile specialmente per la soluzione interattiva dei problemi quando `kubectl exec` è insufficiente perché un container si è arrestato in modo anomalo o un'immagine del container non include utilità di debug. Un esempio di container che include un'utilità di debug è dato dalle immagini senza Distro. Per ulteriori informazioni, consulta Debug con un container di debug temporaneo nella documentazione di Kubernetes.
- Kubernetes ha promosso l'API stabile `HorizontalPodAutoscaler autoscaling/v2` alla disponibilità generale. La API `HorizontalPodAutoscaler autoscaling/v2beta2` è obsoleta. Non sarà disponibile in `1.26`.
- L'opzione `goaway-chance` nel server API Kubernetes aiuta a evitare che le connessioni client `HTTP/2` rimangano bloccate su una singola istanza del server API, chiudendo casualmente una connessione. Quando la connessione viene chiusa, il client tenterà di riconnettersi e probabilmente giungerà su un server API diverso a seguito del sistema di bilanciamento del carico. Amazon EKS versione `1.23` ha abilitato il flag di `goaway-chance`. Se il carico di lavoro in esecuzione sul cluster Amazon EKS utilizza un client non compatibile con `HTTP GOAWAY`, è preferibile aggiornare il client in modo che gestisca `GOAWAY` ristabilendo la connessione quando questa si arresta.

Per il changelog completo di Kubernetes `1.23`, consulta https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG/CHANGELOG-1.23.md#changelog-since-v1220.

Note di rilascio per le versioni 1.21 e 1.22

> **Important**
>
> Non è possibile creare nuovi cluster con queste versioni.

Questo argomento fornisce importanti modifiche di cui tenere conto per le versioni `1.22` e `1.21`. Durante l'aggiornamento, esamina attentamente le modifiche apportate tra la vecchia e la nuova versione del cluster.

Kubernetes versione **`1.22`**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.22`: `DefaultStorageClass`, `DefaultTolerationSeconds`, `LimitRanger`, `MutatingAdmissionWebhook`, `NamespaceLifecycle`, `NodeRestriction`, `ResourceQuota`, `ServiceAccount`, `ValidatingAdmissionWebhook`, `PodSecurityPolicy`, `TaintNodesByCondition`, `StorageObjectInUseProtection`, `PersistentVolumeClaimResize`, `ExtendedResourceToleration`, `CertificateApproval`, `PodPriority`, `CertificateSigning`, `CertificateSubjectRestriction`, `RuntimeClass` e `DefaultIngressClass`.

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
`1.22.17`	`eks.28`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	16 maggio 2024
`1.22.17`	`eks.26`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	1 aprile 2024
`1.22.17`	`eks.14`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	30 giugno 2023

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.22.17	eks.13	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	9 giugno 2023
1.22.17	eks.12	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	5 maggio 2023
1.22.17	eks.11	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	24 marzo 2023
1.22.16	eks.10	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	27 gennaio 2023
1.22.15	eks.9	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	5 dicembre 2022
1.22.15	eks.8	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	18 novembre 2022
1.22.15	eks.7	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	7 novembre 2022
1.22.13	eks.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	21 settembre 2022
1.22.10	eks.5	Nuova versione della piattaforma con resilienza migliorata `etcd`.	15 agosto 2022

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.22.10	eks.4	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti. Questa versione della piattaforma introduce anche un nuovo controller di tagging che etichetta tutti i nodi di lavoro con aws:eks:cluster-name per semplificare l'allocazione dei costi per questi nodi worker. Per ulteriori informazioni, consulta Tagging delle risorse per la fatturazione.	21 luglio 2022
1.22.10	eks.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	7 luglio 2022
1.22.9	eks.2	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	31 maggio 2022
1.22.6	eks.1	Rilascio iniziale di Kubernetes versione 1.22 per Amazon EKS.	4 aprile 2022

Kubernetes versione **1.21**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma 1.21: DefaultStorageClass, DefaultTolerationSeconds, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, ResourceQuota, ServiceAccount, ValidatingAdmissionWebhook, PodSecurityPolicy, TaintNodesByCondition, StorageObjectInUseProtection, PersistentVolumeClaimResize, ExtendedResourceToleration, CertificateApproval, PodPriority, CertificateSigning, CertificateSubjectRestriction, RuntimeClass e DefaultIngressClass.

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.21.14	eks.33	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	16 maggio 2024
1.21.14	eks.31	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	1 aprile 2024
1.21.14	eks.18	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	9 giugno 2023
1.21.14	eks.17	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	5 maggio 2023
1.21.14	eks.16	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	24 marzo 2023
1.21.14	eks.15	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	27 gennaio 2023
1.21.14	eks.14	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	5 dicembre 2022
1.21.14	eks.13	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	18 novembre 2022

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.21.14	eks.12	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	7 novembre 2022
1.21.13	eks.11	Nuova versione della piattaforma con resilienza migliorata etcd.	10 ottobre 2022
1.21.13	eks.10	Nuova versione della piattaforma con resilienza migliorata etcd.	15 agosto 2022
1.21.13	eks.9	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti. Questa versione della piattaforma introduce anche un nuovo controller di tagging che etichetta tutti i nodi di lavoro con aws:eks:cluster-name per semplificare l'allocazione dei costi per questi nodi worker. Per ulteriori informazioni, consulta Tagging delle risorse per la fatturazione.	21 luglio 2022
1.21.13	eks.8	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	7 luglio 2022
1.21.12	eks.7	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	31 maggio 2022

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.21.9	eks.6	L' AWS Security Token Service endpoint viene ripristinato all'endpoint globale dalla versione precedente della piattaforma. Per utilizzare l'endpoint regionale con i ruoli IAM destinati agli account di servizio, è necessario abilitarlo. Per istruzioni su come abilitare l'endpoint regionale, consulta Configurare l' AWS Security Token Service endpoint per un account di servizio.	8 aprile 2022
1.21.5	eks.5	L'impostazione predefinita per i Ruoli IAM per gli account di servizio prevede ora l'utilizzo dell'endpoint regionale AWS Security Token Service , invece dell'endpoint globale. Questa modifica è stata ripristinata, tuttavia, all'endpoint globale in eks.6. Un pianificatore Fargate aggiornato effettua il provision ing dei nodi a una velocità significativamente più elevata durante le implementazioni di grandi dimensioni.	10 marzo 2022

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.21.5	eks.4	La versione 1.10.1-eksbuild.1 del componente aggiuntivo CNI autogestito di Amazon VPC e Amazon EKS è ora la versione predefinita implementata.	13 dicembre 2021
1.21.2	eks.3	Nuova versione della piattaforma con supporto per la gestione degli indirizzi IPv4 di Windows sul controller di risorse VPC in esecuzione sul piano di controllo (control-plane) Kubernetes. Aggiunta la direttiva filtro di Kubernetes per la registrazione di Fargate Fluent Bit.	8 novembre 2021
1.21.2	eks.2	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	17 settembre 2021
1.21.2	eks.1	Rilascio iniziale di Kubernetes versione 1.21 per Amazon EKS.	19 luglio 2021

Versioni della piattaforma Amazon EKS

Le versioni della piattaforma Amazon EKS rappresentano le funzionalità del piano di controllo del cluster. Ad esempio quali flag del server API di Kubernetes sono abilitati e l'attuale versione della patch di Kubernetes. Ogni versione secondaria di Kubernetes dispone di una o più versioni della piattaforma Amazon EKS associate. Le versioni della piattaforma per versioni secondarie di Kubernetes diverse sono indipendenti. Puoi recuperare la versione corrente della piattaforma del

cluster utilizzando AWS CLI o AWS Management Console. Se hai un cluster locale attivo AWS Outposts, consulta Versioni della piattaforma del cluster locale Amazon EKS invece di questo argomento.

Quando una nuova versione Kubernetes secondaria è disponibile in Amazon EKS, ad esempio 1.30, la versione iniziale della piattaforma Amazon EKS per quella versione Kubernetes secondaria inizia `eks.1` da. Tuttavia, Amazon EKS rilascia periodicamente nuove versioni della piattaforma per abilitare nuove impostazioni del piano di controllo Kubernetes e fornire soluzioni per problemi relativi alla sicurezza.

Quando nuove versioni della piattaforma Amazon EKS diventano disponibili per una versione secondaria:

- Il numero di versione della piattaforma Amazon EKS viene incrementato (`eks.`*`n+1`*).
- Amazon EKS aggiorna automaticamente tutti i cluster esistenti alla versione della piattaforma Amazon EKS più recente per la versione secondaria Kubernetes corrispondente. Aggiornamenti automatici di versioni della piattaforma Amazon EKS esistenti vengono implementati in modo incrementale. Il processo di implementazione potrebbe richiedere alcuni minuti. Se le caratteristiche della versione della piattaforma Amazon EKS sono richieste immediatamente, occorre creare un nuovo cluster.

  Se il tuo cluster ha più di due versioni di piattaforma rispetto alla versione attuale della piattaforma, è possibile che Amazon EKS non sia riuscito ad aggiornare automaticamente il cluster. Per i dettagli su cosa potrebbe causare questo problema, consulta La versione della piattaforma Amazon EKS è più avanti di due versioni rispetto all'attuale versione della piattaforma.
- Amazon EKS potrebbe pubblicare un nuovo nodo AMI con una versione patch corrispondente. Tuttavia, tutte le versioni patch sono compatibili tra il piano di controllo EKS e le AMI dei nodi per una data versione secondaria di Kubernetes.

Nuove versioni della piattaforma Amazon EKS non introducono modifiche di conflitto né causano interruzioni del servizio.

I cluster vengono sempre creati con la versione della piattaforma Amazon EKS disponibile più recente (`eks.`*`n`*) per la versione Kubernetes specificata. Se aggiorni il cluster a una nuova versione secondaria Kubernetes, il cluster riceverà la versione della piattaforma Amazon EKS corrente per la versione secondaria Kubernetes a cui è stata aggiornata.

Le versioni della piattaforma Amazon EKS correnti e recenti sono descritte nelle tabelle qui di seguito.

Kubernetes versione **1.30**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.30`: `NodeRestriction`, `ExtendedResourceToleration`, `NamespaceLifecycle`, `LimitRanger`, `ServiceAccount`, `TaintNodesByCondition`, `PodSecurity`, `Priority`, `DefaultTolerationSeconds`, `DefaultStorageClass`, `StorageObjectInUseProtection`, `PersistentVolumeClaimResize`, `RuntimeClass`, `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `MutatingAdmissionWebhook`, `ValidatingAdmissionWebhook`, `ResourceQuota`.

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
`1.30.1`	`eks.3`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	7 giugno 2024
`1.30.0`	`eks.2`	Versione iniziale della versione Kubernetes per EKS. `1.30` Per ulteriori informazioni, consulta Kubernetes1,30.	23 maggio 2024

Kubernetes versione **1.29**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.29`: `NodeRestriction`, `ExtendedResourceToleration`, `NamespaceLifecycle`, `LimitRanger`, `ServiceAccount`, `TaintNodesByCondition`, `PodSecurity`, `Priority`, `DefaultTolerationSeconds`, `DefaultStorageClass`, `StorageObjectInUseProtection`, `PersistentVolumeClaimResize`, `RuntimeClass`, `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `MutatingAdmissionWebhook`, `ValidatingAdmissionWebhook`, `ResourceQuota`.

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.29.5	eks.8	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	7 giugno 2024
1.29.4	eks.7	Nuova versione della piattaforma con scalabilità automatica di CoredNS, correzioni di sicurezza e miglioram enti. Per ulteriori informazioni sull'autoscaling di CoredNS, vedere. Scalabilità automatica CoreDNS	16 maggio 2024
1.29.3	eks.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	18 aprile 2024
1.29.1	eks.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	29 marzo 2024
1.29.1	eks.4	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	20 marzo 2024
1.29.1	eks.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	12 marzo 2024
1.29.0	eks.1	Versione iniziale della versione Kubernetes per EKS. 1.29 Per ulteriori informazioni, consulta Kubernetes1.29.	23 gennaio 2024

Kubernetes versione **1.28**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.28`: `NodeRestriction`, `ExtendedResourceToleration`, `NamespaceLifecycle`, `LimitRanger`, `ServiceAccount`, `TaintNodesByCondition`, `PodSecurity`, `Priority`, `DefaultTolerationSeconds`, `DefaultStorageClass`, `StorageObjectInUseProtection`, `PersistentVolumeClaimResize`, `RuntimeClass`, `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `MutatingAdmissionWebhook`, `ValidatingAdmissionWebhook`, `ResourceQuota`.

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
`1.28.10`	`eks.14`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	7 giugno 2024
`1.28.9`	`eks.13`	Nuova versione della piattaforma con scalabilità automatica di CoredNS, correzioni di sicurezza e miglioram enti. Per ulteriori informazioni sull'autoscaling di CoredNS, vedere. Scalabilità automatica CoreDNS	16 maggio 2024
`1.28.8`	`eks.12`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	18 aprile 2024
`1.28.7`	`eks.11`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	29 marzo 2024
`1.28.7`	`eks.10`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	20 marzo 2024

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.28.6	eks.9	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 marzo 2024
1.28.5	eks.7	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	17 gennaio 2024
1.28.4	eks.6	Nuova versione della piattaforma con voci di accesso, correzioni di sicurezza e miglioramenti.	14 dicembre 2023
1.28.4	eks.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 dicembre 2023
1.28.3	eks.4	Nuova versione della piattaforma con EKS Pod Identity, correzioni di sicurezza e miglioramenti.	10 novembre 2023
1.28.3	eks.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	3 novembre 2023
1.28.2	eks.2	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	16 ottobre 2023
1.28.1	eks.1	Versione iniziale della versione Kubernetes per EKS. 1.28 Per ulteriori informazioni, consulta Kubernetes1.28.	26 settembre 2023

Kubernetes versione **1.27**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.27`: `NodeRestriction`, `ExtendedResourceToleration`, `NamespaceLifecycle`, `LimitRanger`, `ServiceAccount`, `TaintNodesByCondition`, `PodSecurity`, `Priority`, `DefaultTolerationSeconds`, `DefaultStorageClass`, `StorageObjectInUseProtection`, `PersistentVolumeClaimResize`, `RuntimeClass`, `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `MutatingAdmissionWebhook`, `ValidatingAdmissionWebhook`, `ResourceQuota`.

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
`1.27.14`	`eks.18`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	7 giugno 2024
`1.27.13`	`eks.17`	Nuova versione della piattaforma con scalabilità automatica di CoredNS, correzioni di sicurezza e miglioram enti. Per ulteriori informazioni sull'autoscaling di CoredNS, vedere. Scalabilità automatica CoreDNS	16 maggio 2024
`1.27.12`	`eks.16`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	18 aprile 2024
`1.27.11`	`eks.15`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	29 marzo 2024
`1.27.11`	`eks.14`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	20 marzo 2024

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.27.10	eks.13	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 marzo 2024
1.27.9	eks.11	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	17 gennaio 2024
1.27.8	eks.10	Nuova versione della piattaforma con voci di accesso, correzioni di sicurezza e miglioramenti.	14 dicembre 2023
1.27.8	eks.9	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 dicembre 2023
1.27.7	eks.8	Nuova versione della piattaforma con EKS Pod Identity, correzioni di sicurezza e miglioramenti.	10 novembre 2023
1.27.7	eks.7	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	3 novembre 2023
1.27.6	eks.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	16 ottobre 2023
1.27.4	eks.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 agosto 2023
1.27.4	eks.4	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 luglio 2023

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.27.3	eks.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	30 giugno 2023
1.27.2	eks.2	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	9 giugno 2023
1.27.1	eks.1	Versione iniziale della versione Kubernetes per EKS. 1.27 Per ulteriori informazioni, consulta Kubernetes1.27.	24 maggio 2023

Kubernetes versione **1.26**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma 1.26: NodeRestriction, ExtendedResourceToleration, NamespaceLifecycle, LimitRanger, ServiceAccount, TaintNodesByCondition, PodSecurity, Priority, DefaultTolerationSeconds, DefaultStorageClass, StorageObjectInUseProtection, PersistentVolumeClaimResize, RuntimeClass, CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, MutatingAdmissionWebhook, ValidatingAdmissionWebhook, ResourceQuota.

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.26.15	eks.19	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	7 giugno 2024
1.26.15	eks.18	Nuova versione della piattaforma con scalabilità automatica di CoredNS, correzioni di sicurezza e miglioram enti. Per ulteriori informazioni	16 maggio 2024

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
		sull'autoscaling di CoredNS, vedere. Scalabilità automatica CoreDNS	
1.26.15	eks.17	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	18 aprile 2024
1.26.14	eks.16	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	29 marzo 2024
1.26.14	eks.15	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	20 marzo 2024
1.26.13	eks.14	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	12 marzo 2024
1.26.12	eks.12	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	17 gennaio 2024
1.26.11	eks.11	Nuova versione della piattaforma con voci di accesso, correzioni di sicurezza e miglioramenti.	14 dicembre 2023
1.26.11	eks.10	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	12 dicembre 2023
1.26.10	eks.9	Nuova versione della piattaforma con EKS Pod Identity, correzioni di sicurezza e miglioramenti.	10 novembre 2023

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.26.10	eks.8	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	3 novembre 2023
1.26.9	eks.7	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	16 ottobre 2023
1.26.7	eks.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 agosto 2023
1.26.7	eks.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 luglio 2023
1.26.6	eks.4	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 giugno 2023
1.26.5	eks.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	9 giugno 2023
1.26.4	eks.2	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	5 maggio 2023
1.26.2	eks.1	Versione iniziale della versione Kubernetes per EKS. 1.26 Per ulteriori informazioni, consulta Kubernetes1.26.	11 aprile 2023

Kubernetes versione **1.25**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.25`: `NodeRestriction`, `ExtendedResourceToleration`, `NamespaceLifecycle`, `LimitRanger`, `ServiceAccount`, `TaintNodesByCondition`, `PodSecurity`, `Priority`, `DefaultTolerationSeconds`, `DefaultStorageClass`, `StorageObjectInUseProtection`, `PersistentVolumeClaimResize`, `RuntimeClass`, `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `MutatingAdmissionWebhook`, `ValidatingAdmissionWebhook`, `ResourceQuota`.

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
`1.25.16`	`eks.20`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	7 giugno 2024
`1.25.16`	`eks.19`	Nuova versione della piattaforma con scalabilità automatica di CoredNS, correzioni di sicurezza e miglioram enti. Per ulteriori informazioni sull'autoscaling di CoredNS, vedere. Scalabilità automatica CoreDNS	16 maggio 2024
`1.25.16`	`eks.18`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	18 aprile 2024
`1.25.16`	`eks.17`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	29 marzo 2024
`1.25.16`	`eks.16`	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	20 marzo 2024

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.25.16	eks.15	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 marzo 2024
1.25.16	eks.13	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	17 gennaio 2024
1.25.16	eks.12	Nuova versione della piattaforma con voci di accesso, correzioni di sicurezza e miglioramenti.	14 dicembre 2023
1.25.16	eks.11	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 dicembre 2023
1.25.15	eks.10	Nuova versione della piattaforma con EKS Pod Identity, correzioni di sicurezza e miglioramenti.	10 novembre 2023
1.25.15	eks.9	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	3 novembre 2023
1.25.14	eks.8	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	16 ottobre 2023
1.25.12	eks.7	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 agosto 2023
1.25.12	eks.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 luglio 2023

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.25.11	eks.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 giugno 2023
1.25.10	eks.4	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	9 giugno 2023
1.25.9	eks.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	5 maggio 2023
1.25.8	eks.2	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	24 marzo 2023
1.25.6	eks.1	Versione iniziale della versione Kubernetes per EKS. 1.25 Per ulteriori informazioni, consulta Kubernetes1,25.	21 febbraio 2023

Kubernetes versione **1.24**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma 1.24: CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, Priority, PodSecurityPolicy, ResourceQuota, RuntimeClass, ServiceAccount, StorageObjectInUseProtection, TaintNodesByCondition e ValidatingAdmissionWebhook.

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.24.17	eks.23	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	7 giugno 2024
1.24.17	eks.22	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	16 maggio 2024
1.24.17	eks.21	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	18 aprile 2024
1.24.17	eks.20	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	29 marzo 2024
1.24.17	eks.19	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	20 marzo 2024
1.24.17	eks.18	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 marzo 2024
1.24.17	eks.16	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	17 gennaio 2024
1.24.17	eks.15	Nuova versione della piattaforma con voci di accesso, correzioni di sicurezza e miglioramenti.	14 dicembre 2023
1.24.17	eks.14	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 dicembre 2023

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.24.17	eks.13	Nuova versione della piattaforma con EKS Pod Identity, correzioni di sicurezza e miglioramenti.	10 novembre 2023
1.24.17	eks.12	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	3 novembre 2023
1.24.17	eks.11	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	16 ottobre 2023
1.24.16	eks.10	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	30 agosto 2023
1.24.16	eks.9	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	30 luglio 2023
1.24.15	eks.8	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	30 giugno 2023
1.24.14	eks.7	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	9 giugno 2023
1.24.13	eks.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	5 maggio 2023
1.24.12	eks.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	24 marzo 2023

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.24.8	eks.4	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	27 gennaio 2023
1.24.7	eks.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	5 dicembre 2022
1.24.7	eks.2	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	18 novembre 2022
1.24.7	eks.1	Versione iniziale della versione Kubernetes per EKS. 1.24 Per ulteriori informazioni, consulta Kubernetes1.24.	15 novembre 2022

Kubernetes versione **1.23**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma 1.23: CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, Priority, PodSecurityPolicy, ResourceQuota, RuntimeClass, ServiceAccount, StorageObjectInUseProtection, TaintNodesByCondition e ValidatingAdmissionWebhook.

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.23.17	eks.25	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	7 giugno 2024

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.23.17	eks.24	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	16 maggio 2024
1.23.17	eks.23	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	18 aprile 2024
1.23.17	eks.22	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	29 marzo 2024
1.23.17	eks.21	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	20 marzo 2024
1.23.17	eks.20	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 marzo 2024
1.23.17	eks.18	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	17 gennaio 2024
1.23.17	eks.17	Nuova versione della piattaforma con voci di accesso, correzioni di sicurezza e miglioramenti.	14 dicembre 2023
1.23.17	eks.16	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	12 dicembre 2023
1.23.17	eks.15	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	10 novembre 2023

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.23.17	eks.14	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	3 novembre 2023
1.23.17	eks.13	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	16 ottobre 2023
1.23.17	eks.12	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 agosto 2023
1.23.17	eks.11	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 luglio 2023
1.23.17	eks.10	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	30 giugno 2023
1.23.17	eks.9	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	9 giugno 2023
1.23.17	eks.8	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	5 maggio 2023
1.23.17	eks.7	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	24 marzo 2023
1.23.14	eks.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	27 gennaio 2023

Versione di Kubernetes	Versioni della piattaforma EKS	Note di rilascio	Data di rilascio
1.23.13	eks.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	5 dicembre 2022
1.23.13	eks.4	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	18 novembre 2022
1.23.12	eks.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	7 novembre 2022
1.23.10	eks.2	Nuova versione della piattaforma con correzioni di sicurezza e miglioram enti.	21 settembre 2022
1.23.7	eks.1	Versione iniziale della versione Kubernetes per EKS. 1.23 Per ulteriori informazioni, consulta Kubernetes1.23.	11 agosto 2022

Ottieni la versione corrente della piattaforma

Per ottenere la versione corrente della piattaforma per il tuo cluster (console)

1. Aprire la Console Amazon EKS.
2. Nel pannello di navigazione scegliere Clusters (Cluster).
3. Nell'elenco dei cluster, scegli il nome del cluster di cui verificare la versione della piattaforma.
4. Seleziona la scheda Panoramica.
5. La versione della piattaforma è disponibile nella sezione Dettagli.

Per ottenere la versione corrente della piattaforma per il tuo cluster (AWS CLI)

1. Determina il nome del cluster di cui vuoi controllare la versione della piattaforma.
2. Esegui il comando seguente:

```
aws eks describe-cluster --name my-cluster --query cluster.platformVersion
```

Di seguito viene riportato un output di esempio:

```
"eks.10"
```

Scalabilità automatica

Il dimensionamento automatico è una funzione che dimensiona automaticamente in verticale e in orizzontale le risorse per soddisfare le esigenze in continua evoluzione. Questa è una delle principali funzioni Kubernetes, la cui esecuzione manuale richiederebbe altrimenti diverse risorse umane.

Amazon EKS supporta due prodotti con dimensionamento automatico:

Karpenter

Karpenter è una soluzione di dimensionamento automatico dei cluster Kubernetes, flessibile e ad alte prestazioni, che aiuta a migliorare la disponibilità delle applicazioni e l'efficienza del cluster. Karpenter avvia risorse di calcolo della giusta dimensione (ad esempio, istanze Amazon EC2) in risposta alla modifica del carico delle applicazioni in meno di un minuto. Tramite l'integrazione di Kubernetes con AWS, Karpenter è in grado di fornire risorse di calcolo just-in-time che soddisfano con precisione i requisiti del carico di lavoro. Karpenter effettua automaticamente il provisioning di nuove risorse di calcolo in base ai requisiti specifici dei carichi di lavoro del cluster. Questi includono requisiti di calcolo, archiviazione, accelerazione e pianificazione. Amazon EKS supporta i cluster che utilizzano Karpenter, anche se Karpenter funziona con qualsiasi cluster Kubernetes conforme. Per ulteriori informazioni, consulta la documentazione Karpenter.

Cluster Autoscaler

Il Cluster Autoscaler di Kubernetes regola automaticamente il numero di nodi nel cluster quando i pod riscontrano un errore o vengono riprogrammati su altri nodi. Il Cluster Autoscaler utilizza i gruppi con scalabilità automatica. Per ulteriori informazioni consultare Cluster Autoscaler su AWS.

Gestisci l'accesso

Scopri come gestire l'accesso al tuo cluster Amazon EKS. L'utilizzo di Amazon EKS richiede la conoscenza di come entrambi Kubernetes e AWS Identity and Access Management (AWS IAM) gestiscono il controllo degli accessi.

Questa sezione include:

the section called “Concedi l'accesso alle API Kubernetes”— Scopri come consentire alle applicazioni o agli utenti di autenticarsi nell'KubernetesAPI. Puoi utilizzare le voci di accesso, aws-auth o un provider ConfigMap OIDC esterno.

the section called “Accedi al mio cluster con kubectl”— Scopri come configurare kubectl per comunicare con il tuo cluster Amazon EKS. Usa la AWS CLI per creare un file kubeconfig.

the section called “Concedi l'accesso ai carichi di lavoro a AWS”— Scopri come associare un account di Kubernetes servizio a IAM Roles. AWS Puoi utilizzare Pod Identity o IAM Roles for Service Accounts (IRSA).

Attività comuni:

- Concedi agli sviluppatori l'accesso all'KubernetesAPI. Visualizza Kubernetes le risorse in AWS Management Console.
  - Soluzione: utilizza Access Entries per associare le autorizzazioni Kubernetes RBAC agli utenti o ai ruoli AWS IAM.
- Configura kubectl per comunicare con un cluster Amazon EKS utilizzando le credenziali. AWS
  - Soluzione: utilizzate la AWS CLI per creare un file kubeconfig.
- Utilizza un provider di identità esterno, come Ping Identity, per autenticare gli utenti nell'API. Kubernetes
  - Soluzione: collega un provider OIDC esterno.
- Concedi ai carichi di lavoro sul tuo Kubernetes cluster la possibilità di chiamare le API. AWS
  - Soluzione: utilizza Pod Identity per associare un ruolo AWS IAM a un account di Kubernetes servizio.

Sfondo:

- Scopri come funzionano gli account di servizio Kubernetes.

- Rivedi il modello Kubernetes Role Based Access Control (RBAC)
- Per ulteriori informazioni sulla gestione dell'accesso alle AWS risorse, consulta la IAM User Guide.AWS In alternativa, segui un corso introduttivo gratuito sull'uso AWS di IAM.

Concedi l'accesso alle Kubernetes API

Il tuo cluster ha un endpoint Kubernetes API. Kubectl utilizza questa API. Puoi autenticarti su questa API usando due tipi di identità:

- Un principale AWS Identity and Access Management (IAM) (ruolo o utente): questo tipo richiede l'autenticazione su IAM. Gli utenti possono accedere AWS come utenti IAM o con un'identità federata utilizzando le credenziali fornite tramite una fonte di identità. Gli utenti possono accedere solo con un'identità federata se l'amministratore ha configurato in precedenza la federazione delle identità utilizzando i ruoli IAM. Quando gli utenti accedono AWS utilizzando la federazione, assumono indirettamente un ruolo. Quando gli utenti utilizzano questo tipo di identità:
  - Puoi assegnare loro autorizzazioni Kubernetes in modo che possano lavorare con gli oggetti Kubernetes sul tuo cluster. Per ulteriori informazioni su come assegnare le autorizzazioni ai principali IAM in modo che possano accedere agli oggetti Kubernetes sul cluster, consulta la sezione Gestire le voci di accesso.
  - Puoi assegnare loro le autorizzazioni IAM in modo che possano lavorare con il tuo cluster Amazon EKS e le sue risorse utilizzando l'API Amazon EKS, AWS CLI, AWS CloudFormation AWS Management Console, o. `eksctl` Per ulteriori informazioni, consulta Actions defined by Amazon Elastic Kubernetes Service nella Documentazione di riferimento per l'autorizzazione ai servizi.
  - I nodi si aggiungono al cluster assumendo un ruolo IAM. La capacità di accedere al cluster utilizzando i principali (IAM) è abilitata dall'autenticatore AWS IAM per Kubernetes, che viene eseguito sul piano di controllo di Amazon EKS.
- Un utente del tuo provider OpenID Connect (OIDC): questo tipo richiede l'autenticazione al tuo provider OIDC. Per ulteriori informazioni sulla configurazione del provider OIDC con il cluster Amazon EKS, consulta la sezione Autentica gli utenti del tuo cluster da un provider di OpenID Connect identità. Quando gli utenti utilizzano questo tipo di identità:
  - Puoi assegnare loro autorizzazioni Kubernetes in modo che possano lavorare con gli oggetti Kubernetes sul tuo cluster.

- Non puoi assegnare loro autorizzazioni IAM in modo che possano lavorare con il tuo cluster Amazon EKS e le sue risorse utilizzando l'API Amazon EKS,, AWS CLI AWS CloudFormation AWS Management Console, o. `eksctl`

Puoi utilizzare entrambi i tipi di identità con il tuo cluster. Il metodo di autenticazione IAM non può essere disabilitato. Il metodo di autenticazione OIDC è facoltativo.

Associa le identità IAM alle autorizzazioni Kubernetes

L'Autenticatore AWS IAM per Kubernetes è installato sul piano di controllo del tuo cluster. Abilita i principali AWS Identity and Access Management (IAM) (ruoli e utenti) a cui consenti di accedere alle risorse Kubernetes sul tuo cluster. Puoi consentire ai principali IAM di accedere agli oggetti Kubernetes sul tuo cluster utilizzando uno dei metodi seguenti:

- Creazione di voci di accesso: se la versione del cluster corrisponde o è successiva alla versione della piattaforma elencata nella sezione Prerequisiti per la versione di Kubernetes del cluster, ti consigliamo di utilizzare questa opzione.

  Utilizza le voci di accesso per gestire le autorizzazioni Kubernetes dei principali IAM dall'esterno del cluster. Puoi aggiungere e gestire l'accesso al cluster utilizzando l'API EKS AWS Command Line Interface, AWS gli SDK e. AWS CloudFormation AWS Management Console Ciò significa che puoi gestire gli utenti con gli stessi strumenti con cui hai creato il cluster.

  Per iniziare, segui Configurazione delle voci di accesso, quindi Migrazione delle voci `aws-auth ConfigMap` esistenti alle voci di accesso.
- Aggiunta di voci a **`aws-auth ConfigMap`**: se la versione della piattaforma del cluster è precedente alla versione elencata nella sezione Prerequisiti, è necessario utilizzare questa opzione. Se la versione della piattaforma del cluster corrisponde o è successiva alla versione della piattaforma elencata nella sezione Prerequisiti per la versione di Kubernetes del cluster e hai aggiunto delle voci a `ConfigMap`, ti consigliamo di migrare tali voci per potervi accedere. Tuttavia, non puoi migrare le voci aggiunte da Amazon EKS a `ConfigMap`, come le voci per i ruoli IAM utilizzati con gruppi di nodi gestiti o profili Fargate. Per ulteriori informazioni, consulta the section called “Concedi l'accesso alle API Kubernetes”.
  - Se è necessario utilizzare l'opzione `aws-auth ConfigMap`, è possibile aggiungere voci a `ConfigMap` utilizzando il comando **`eksctl create iamidentitymapping`**. Per ulteriori informazioni, consulta Manage IAM users and roles nella documentazione di `eksctl`.

Imposta la modalità di autenticazione del cluster

Ogni cluster dispone di una modalità di autenticazione. La modalità di autenticazione determina quali metodi è possibile utilizzare per consentire ai principali IAM di accedere agli oggetti Kubernetes sul cluster. Sono disponibili tre modalità di autenticazione.

Important

Una volta abilitato, il metodo di immissione dell'accesso non può essere disabilitato. Se il `ConfigMap` metodo non è abilitato durante la creazione del cluster, non può essere abilitato in seguito. Tutti i cluster creati prima dell'introduzione delle voci di accesso hanno il `ConfigMap` metodo abilitato.

`aws-auth` `ConfigMap` all'interno del cluster

Questa è la modalità di autenticazione originale per i cluster Amazon EKS. Il principale IAM che ha creato il cluster è l'utente iniziale che può accedere al cluster utilizzando `kubectl`. L'utente iniziale deve aggiungere altri utenti all'elenco in `aws-auth` `ConfigMap` e assegnare autorizzazioni che influiscono sugli altri utenti all'interno del cluster. Questi altri utenti non possono gestire o rimuovere l'utente iniziale, poiché non c'è una voce da gestire in `ConfigMap`.

Sia `ConfigMap` sia voci di accesso

Con questa modalità di autenticazione, è possibile utilizzare entrambi i metodi per aggiungere i principali IAM al cluster. Si noti che ogni metodo memorizza voci separate; ad esempio, se si aggiunge una voce di accesso da AWS CLI, non `aws-auth` `ConfigMap` viene aggiornata.

Solo voci di accesso

Con questa modalità di autenticazione, puoi utilizzare l'API EKS e AWS gli SDK e gestire l'accesso AWS Management Console al cluster per i principali IAM. AWS Command Line Interface AWS CloudFormation

Ogni voce di accesso include un tipo, che consente di utilizzare la combinazione di un ambito di accesso per limitare il principale a uno spazio dei nomi specifico e una policy di accesso per definire policy di autorizzazione riutilizzabili preconfigurate. In alternativa, è possibile utilizzare il tipo STANDARD e i gruppi RBAC di Kubernetes per assegnare autorizzazioni personalizzate.

Modalità di autenticazione	Metodi
Solo `ConfigMap` (`CONFIG_MAP`)	`aws-auth ConfigMap`
API EKS e `ConfigMap` (`API_AND_CONFIG_MAP`)	accedere alle voci nell'API EKS AWS Command Line Interface, negli AWS SDK e e AWS CloudFormation AWS Management Console `aws-auth ConfigMap`
Solo API EKS (`API`)	accedere alle voci nell'API EKS AWS Command Line Interface, negli AWS SDK e AWS CloudFormation AWS Management Console

Gestire le voci di accesso

Prerequisiti

- Familiarità con le opzioni di accesso al cluster per il cluster Amazon EKS. Per ulteriori informazioni, consulta Concedi l'accesso alle Kubernetes API .
- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS. Per utilizzare le voci di accesso e cambiare la modalità di autenticazione di un cluster, il cluster deve avere una versione della piattaforma corrispondente o successiva a quella elencata nella tabella seguente o una versione di Kubernetes successiva a quelle elencate nella tabella.

Versione Kubernetes	Versione della piattaforma
`1.30`	`eks.2`
`1.29`	`eks.1`
`1.28`	`eks.6`
`1.27`	`eks.10`
`1.26`	`eks.11`

Versione Kubernetes	Versione della piattaforma
1.25	eks.12
1.24	eks.15
1.23	eks.17

È possibile verificare la versione corrente di Kubernetes e della piattaforma sostituendo *my-cluster* nel seguente comando con il nome del cluster e quindi eseguendo il comando modificato: **aws eks describe-cluster --name *my-cluster* --query 'cluster.{"Kubernetes Version": version, "Platform Version": platformVersion}'**.

> **Important**
>
> Dopo che Amazon EKS ha aggiornato il cluster alla versione della piattaforma elencata nella tabella, Amazon EKS crea una voce di accesso con autorizzazioni di amministratore al cluster per il principale IAM che ha originariamente creato il cluster. Se non desideri che il principale IAM disponga delle autorizzazioni di amministratore per il cluster, rimuovi la voce di accesso creata da Amazon EKS.
>
> Per i cluster con versioni della piattaforma precedenti a quelle elencate nella tabella precedente, il creatore del cluster è sempre un amministratore del cluster. Non è possibile rimuovere le autorizzazioni di amministratore del cluster dall'utente o dal ruolo IAM che ha creato il cluster.

- Un principale IAM con le seguenti autorizzazioni per il cluster: `CreateAccessEntry`, `ListAccessEntries`, `DescribeAccessEntry`, `DeleteAccessEntry` e `UpdateAccessEntry`. Per ulteriori informazioni sulle autorizzazioni Amazon EKS, consulta Actions defined by Amazon Elastic Kubernetes Service nella Documentazione di riferimento per l'autorizzazione ai servizi.
- Un principale IAM esistente per cui creare una voce di accesso o una voce di accesso esistente da aggiornare o eliminare.

Configurazione delle voci di accesso

Per iniziare a utilizzare le voci di accesso, è necessario modificare la modalità di autenticazione del cluster nelle modalità `API_AND_CONFIG_MAP` o `API`. Questa operazione aggiunge l'API per le voci di accesso.

AWS Management Console

Per creare una voce di accesso

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli il nome del cluster in cui desideri creare una voce di accesso.
3. Scegli la scheda Accesso.
4. Modalità di autenticazione mostra la modalità corrente di autenticazione del cluster. Se la modalità mostra EKS API, puoi già aggiungere voci di accesso e puoi saltare i passaggi rimanenti.
5. Scegli Gestisci accesso.
6. Per Modalità di autenticazione del cluster, seleziona una modalità con l'EKS API. Tieni presente che non puoi ripristinare la modalità di autenticazione in una modalità che rimuove EKS API e le voci di accesso.
7. Seleziona Salvataggio delle modifiche. Amazon EKS inizia ad aggiornare il cluster, lo stato del cluster cambia in Updating e la modifica viene registrata nella scheda Cronologia degli aggiornamenti.
8. Attendi che lo stato del cluster ritorni su Active. Quando il cluster è Active, puoi seguire i passaggi indicati in Creazione di voci di accesso per aggiungere l'accesso al cluster per i principali IAM.

AWS CLI

Prerequisito

L'ultima versione della AWS CLI v1 installata e configurata sul tuo dispositivo o. AWS CloudShell AWS CLI la v2 non supporta nuove funzionalità per alcuni giorni. È possibile verificare la versione corrente con `aws --version | cut -d / -f2 | cut -d ' ' -f1`. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta Installazione,

aggiornamento e disinstallazione AWS CLI e Configurazione rapida con `aws configure` nella Guida per l' AWS Command Line Interface utente. La AWS CLI versione installata in AWS CloudShell può anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l' AWS CloudShell utente.

1.
2. Esegui il comando seguente. Sostituisci *my-cluster* con il nome del tuo cluster. Se desideri disabilitare il metodo `ConfigMap` in modo permanente, sostituisci `API_AND_CONFIG_MAP` con `API`.

   Amazon EKS inizia ad aggiornare il cluster, lo stato del cluster cambia in UPDATING e la modifica viene registrata in aws eks list-updates.

   ```
   aws eks update-cluster-config --name my-cluster --access-config
    authenticationMode=API_AND_CONFIG_MAP
   ```
3. Attendi che lo stato del cluster ritorni su Active. Quando il cluster è Active, puoi seguire i passaggi indicati in Creazione di voci di accesso per aggiungere l'accesso al cluster per i principali IAM.

Creazione di voci di accesso

Considerazioni

Prima di creare voci di accesso, considera quanto segue:

- Una voce di accesso include il nome della risorsa Amazon (ARN) di un principale (e solo di uno) IAM esistente. Un principale IAM non può essere incluso in più di una voce di accesso. Considerazioni aggiuntive per l'ARN specificato:
  - Le best practice IAM consigliano di accedere al cluster utilizzando ruoli IAM con credenziali a breve termine, anziché utenti IAM con credenziali a lungo termine. Per ulteriori informazioni, consulta Richiedere agli utenti umani di utilizzare la federazione con un provider di identità per accedere AWS utilizzando credenziali temporanee nella Guida per l'utente IAM.
  - Se l'ARN è per un ruolo IAM, può includere un percorso. Gli ARN nelle voci `aws-auth ConfigMap`, non possono includere un percorso. Ad esempio, il tuo ARN può essere `arn:aws:iam::`*`111122223333`*`:role/`*`development/apps/my-role`* o `arn:aws:iam::`*`111122223333`*`:role/`*`my-role`*.

- Se il tipo di voce di accesso è diverso da `STANDARD` (vedi la prossima considerazione sui tipi), l'ARN deve trovarsi nello stesso in Account AWS cui si trova il cluster. Se il tipo è`STANDARD`, l'ARN può essere uguale o diverso dall'account in cui si trova il cluster. Account AWS
- Non è possibile modificare il principale IAM dopo aver creato la voce di accesso.
- Se elimini il principale IAM con questo ARN, la voce di accesso non viene eliminata automaticamente. Ti consigliamo di eliminare la voce di accesso con un ARN per un principale IAM che elimini. Se non cancelli la voce di accesso e in futuro ricrei il principale IAM, anche se ha lo stesso ARN, la voce di accesso non funzionerà. Questo perché, anche se l'ARN è lo stesso per il principale IAM ricreato, l'`roleID`or `userID` (puoi vederlo con il `aws sts get-caller-identity` AWS CLI comando) è diverso per il principale IAM ricreato rispetto al principale IAM originale. Anche se non visualizzi il `roleID` o il `userID` del principale IAM per una voce di accesso, Amazon EKS lo memorizza insieme alla voce di accesso.

- Ogni voce di accesso include un tipo. Puoi specificare `EC2 Linux` (per un ruolo IAM utilizzato con nodi autogestiti Linux o Bottlerocket), `EC2 Windows` (per un ruolo IAM utilizzato con nodi autogestiti Windows), `FARGATE_LINUX` (per un ruolo IAM utilizzato con AWS Fargate (Fargate)) o come tipo. `STANDARD` Se non specifichi un tipo, Amazon EKS lo imposta automaticamente su `STANDARD`. Non è necessario creare una voce di accesso per un ruolo IAM utilizzato per un gruppo di nodi gestiti o un profilo Fargate, in quanto Amazon EKS aggiunge voci per questi ruoli a `aws-auth` `ConfigMap`, indipendentemente dalla versione della piattaforma in cui si trova il cluster.

  Non è possibile modificare il tipo dopo aver creato la voce di accesso.
- Se il tipo di voce di accesso è `STANDARD`, puoi specificare un nome utente per la voce di accesso. Se non specifichi un valore per il nome utente, Amazon EKS assegnerà automaticamente uno dei valori seguenti, in funzione del tipo di voce di accesso e del principale IAM da te indicato, che si tratti di un ruolo IAM o di un utente IAM. A meno che tu non abbia un motivo particolare per specificare il tuo nome utente, consigliamo di non specificarne uno e di lasciare che Amazon EKS lo generi automaticamente. Se decidi di specificare il tuo nome utente:
  - Tieni presente che non può iniziare con `system:`, `eks:`, `aws:`, `amazon:` o `iam:`.
  - Se il nome utente è per un ruolo IAM, ti consigliamo di aggiungere `{{SessionName}}` alla fine del nome utente. Se lo aggiungi `{{SessionName}}` al tuo nome utente, il nome utente deve includere i due punti prima di {{}}. SessionName Quando si assume questo ruolo, il nome della sessione specificato quando si assume il ruolo viene passato automaticamente al cluster e verrà visualizzato nei CloudTrail log. Ad esempio, non puoi avere un nome utente del tipo `john{{SessionName}}`. Il nome utente deve essere `:john{{SessionName}}` o `jo:hn{{SessionName}}`. Solo i due punti devono essere prima di `{{SessionName}}`. Il

nome utente generato da Amazon EKS nella tabella seguente include un ARN. Poiché un ARN include i due punti, soddisfa questo requisito. I due punti non sono obbligatori se non includi `{{SessionName}}` nel nome utente.

Il tipo di principale IAM	Type	Valore del nome utente che Amazon EKS imposta automaticamente
Utente	`STANDARD`	L'ARN dell'utente. Esempio: `arn:aws:iam::` *`111122223333`* `:user/`*`my-user`*
Ruolo	`STANDARD`	L'ARN STS del ruolo quando viene assunto. Amazon EKS aggiunge `{{SessionName}}` al ruolo. Esempio: `arn:aws:sts::` *`111122223333`* `:assumed-role/`*`my-role`*`/{{SessionName}}` Se l'ARN del ruolo che hai specificato conteneva un percorso, Amazon EKS lo rimuove nel nome utente generato.
Ruolo	`EC2 Linux` o `EC2 Windows`	`system:node:{{EC2PrivateDNSName}}`
Ruolo	`FARGATE_LINUX`	`system:node:{{SessionName}}`

Puoi modificare il nome utente dopo aver creato la voce di accesso.

- Se il tipo di una voce di accesso è `STANDARD` e desideri utilizzare l'autorizzazione RBAC di Kubernetes, puoi aggiungere uno o più nomi dei gruppi alla voce di accesso. Dopo aver creato una voce di accesso, puoi aggiungere e rimuovere i nomi dei gruppi. Affinché il principale IAM abbia accesso agli oggetti Kubernetes sul cluster, è necessario creare e gestire oggetti del controllo degli accessi basato sui ruoli (RBAC) in Kubernetes. Crea `RoleBinding` di Kubernetes o oggetti `ClusterRoleBinding` sul tuo cluster che specifichino il nome del gruppo come `subject` per `kind: Group`. Kubernetes autorizza l'accesso per il principale IAM a tutti gli oggetti del cluster che hai specificato in un `Role` di Kubernetes o un oggetto `ClusterRole` che hai specificato anche nei `roleRef` delle tue associazioni. Se specifichi i nomi dei gruppi, ti consigliamo di acquisire familiarità con gli oggetti del controllo degli accessi basato sui ruoli (RBAC) in Kubernetes. Per ulteriori informazioni, consulta Utilizzo dell'autorizzazione RBAC nella documentazione di Kubernetes.

  > **Important**
  >
  > Amazon EKS non conferma che gli oggetti RBAC di Kubernetes esistenti sul cluster includano i nomi dei gruppi specificati.

  Puoi collegare policy di accesso Amazon EKS a una voce di accesso, come alternativa o complemento all'autorizzazione concessa da Kubernetes al principale IAM per accedere agli oggetti Kubernetes sul tuo cluster. Amazon EKS autorizza i principali IAM ad accedere agli oggetti Kubernetes sul tuo cluster con le autorizzazioni previste dalla policy di accesso. Puoi definire l'ambito delle autorizzazioni di una policy di accesso agli spazi dei nomi di Kubernetes che specifichi. L'uso delle policy di accesso non richiede la gestione degli oggetti RBAC di Kubernetes. Per ulteriori informazioni, consulta Associazione e dissociazione delle policy di accesso alle/dalle voci di accesso.
- Se crei una voce di accesso con tipo `EC2 Linux` o `EC2 Windows`, il principale IAM che crea la voce di accesso deve disporre dell'autorizzazione `iam:PassRole`. Per ulteriori informazioni, consulta Granting a user permissions to pass a role to an Servizio AWS nella Guida per l'utente IAM.
- Analogamente al comportamento di IAM standard, la creazione e gli aggiornamenti delle voci di accesso alla fine sono coerenti e potrebbero essere necessari alcuni secondi prima che la chiamata API iniziale risulti completata con successo. È necessario progettare le applicazioni in modo da tenere in considerazione questi potenziali ritardi Si consiglia di non includere creazioni o aggiornamenti degli accessi nei percorsi critici e ad alta disponibilità del codice dell'applicazione.

Al contrario, apportare modifiche in un'inizializzazione separata o in una routine di configurazione che si esegue meno frequentemente. Inoltre, assicurarsi di verificare che le modifiche siano state propagate prima che i flussi di lavoro di produzione dipendano da esse.

- Le voci di accesso non supportano i ruoli collegati al servizio. Non è possibile creare voci di accesso in cui l'ARN principale è un ruolo collegato al servizio. È possibile identificare i ruoli collegati al servizio in base al relativo ARN, che è nel formato. `arn:aws:iam::*:role/aws-service-role/*`

È possibile creare una voce di accesso utilizzando AWS Management Console o il AWS CLI.

AWS Management Console

Per creare una voce di accesso

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli il nome del cluster in cui desideri creare una voce di accesso.
3. Scegli la scheda Accesso.
4. Seleziona Crea voce di accesso.
5. Per Principale IAM, seleziona un ruolo o un utente IAM esistente. Le best practice IAM consigliano di accedere al cluster utilizzando ruoli IAM con credenziali a breve termine, anziché utenti IAM con credenziali a lungo termine. Per ulteriori informazioni, consulta Richiedere agli utenti umani di utilizzare la federazione con un provider di identità per accedere AWS utilizzando credenziali temporanee nella Guida per l'utente IAM.
6. Per Tipo, se la voce di accesso è per il ruolo del nodo utilizzato per i nodi Amazon EC2 autogestiti, seleziona EC2 Linux o EC2 Windows. Altrimenti, accetta l'impostazione predefinita (Standard).
7. Se il Tipo che hai scelto è Standard e desideri specificare un Nome utente, inserisci il nome utente.
8. Se il Tipo che hai scelto è Standard e desideri utilizzare l'autorizzazione RBAC di Kubernetes per il principale IAM, specifica uno o più nomi per i Gruppi. Se non specifichi nomi dei gruppi e desideri utilizzare l'autorizzazione Amazon EKS, puoi associare una policy di accesso in un passaggio successivo o dopo la creazione della voce di accesso.
9. (Facoltativo) Per Tag, assegna etichette alla voce di accesso. Ad esempio, per facilitare la ricerca di tutte le risorse con lo stesso tag.

10. Seleziona Successivo.
11. Nella pagina Aggiungi policy di accesso, se il tipo che hai scelto era Standard e desideri che Amazon EKS autorizzi il principale IAM a disporre delle autorizzazioni per gli oggetti Kubernetes sul tuo cluster, completa i seguenti passaggi. Altrimenti, scegli Next (Successivo).

    a. Per Nome della policy, scegli una policy di accesso. Non puoi visualizzare le autorizzazioni delle policy di accesso, tuttavia includono autorizzazioni simili a quelle degli oggetti `ClusterRole` di Kubernetes rivolti all'utente. Per ulteriori informazioni, consulta User-facing roles nella documentazione di Kubernetes.

    b. Selezionare una delle seguenti opzioni:

    - Cluster: scegli questa opzione se desideri che Amazon EKS autorizzi il principale IAM a disporre delle autorizzazioni nella policy di accesso per tutti gli oggetti Kubernetes sul tuo cluster.
    - Spazio dei nomi Kubernetes: scegli questa opzione se desideri che Amazon EKS autorizzi il principale IAM a disporre delle autorizzazioni nella policy di accesso per tutti gli oggetti Kubernetes in uno spazio dei nomi Kubernetes specifico sul tuo cluster. Per Spazio dei nomi, inserisci il nome dello spazio dei nomi Kubernetes sul tuo cluster. Se desideri aggiungere altri spazi dei nomi, scegli Aggiungi nuovo spazio dei nomi e inserisci il nome dello spazio dei nomi.

    c. Se desideri aggiungere ulteriori policy, scegli Aggiungi policy. Puoi definire l'ambito di ciascuna policy in modo diverso, ma puoi aggiungere ogni policy solo una volta.

    d. Seleziona Successivo.
12. Verifica la configurazione per la tua voce di accesso. Se noti qualcosa di errato, scegli Precedente per tornare indietro e correggere l'errore. Se la configurazione è corretta, scegli Crea.

AWS CLI

Prerequisito

L'ultima versione della AWS CLI v1 installata e configurata sul tuo dispositivo o. AWS CloudShell AWS CLI la v2 non supporta nuove funzionalità per alcuni giorni. È possibile verificare la versione corrente con `aws --version | cut -d / -f2 | cut -d ' ' -f1`. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta Installazione, aggiornamento e disinstallazione AWS CLI e Configurazione rapida con `aws configure`

nella Guida per l' AWS Command Line Interface utente. La AWS CLI versione installata in AWS CloudShell può anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l' AWS CloudShell utente.

Per creare una voce di accesso

Per creare voci di accesso è possibile utilizzare uno qualsiasi degli esempi seguenti:

- Crea una voce di accesso per un gruppo di nodi Amazon EC2 Linux autogestito. Sostituisci *my-cluster* con il nome del cluster, *111122223333* con il tuo Account AWS ID e *EKS-My-Cluster-Self-Managed-NG-1* con il nome del ruolo IAM del tuo nodo. Nel caso in cui il tuo gruppo di nodi operi su Windows, sostituisci *EC2_Linux* con **EC2_Windows**.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1 --type EC2_Linux
```

  Non puoi utilizzare l'opzione `--kubernetes-groups` quando specifichi un tipo diverso da `STANDARD`. Non puoi associare una policy di accesso a questa voce di accesso, perché il suo tipo è un valore diverso da `STANDARD`.

- Crea una voce di accesso che permetta a un ruolo IAM, non utilizzato per un gruppo di nodi autogestiti di Amazon EC2, di essere autorizzato da Kubernetes per accedere al tuo cluster. *Sostituisci my-cluster con il nome del tuo cluster, 111122223333 con il tuo ID e my-role con il nome del tuo ruolo IAM. Account AWS* Sostituisci *Visualizzatori* con il nome di un gruppo che hai specificato in un oggetto `ClusterRoleBinding` o `RoleBinding` di Kubernetes sul tuo cluster.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role --type STANDARD --user Viewers --
kubernetes-groups Viewers
```

- Crea una voce di accesso che consenta a un utente IAM di autenticarsi nel tuo cluster. Questo esempio viene fornito perché è possibile, anche se le best practice IAM consigliano di accedere al cluster utilizzando ruoli IAM con credenziali a breve termine, anziché utenti IAM con credenziali a lungo termine. Per ulteriori informazioni, consulta Richiedere agli utenti umani di utilizzare la federazione con un provider di identità per accedere AWS utilizzando credenziali temporanee nella Guida per l'utente IAM.

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:user/my-user --type STANDARD --username my-user
```

Se desideri che questo utente abbia un accesso al tuo cluster maggiore rispetto alle autorizzazioni nei ruoli di rilevamento delle API Kubernetes, è necessario associare una policy di accesso alla voce di accesso, poiché l'opzione `--kubernetes-groups` non viene utilizzata. Per ulteriori informazioni, consulta Associazione e dissociazione delle policy di accesso alle/dalle voci di accesso e API discovery roles nella documentazione di Kubernetes.

Aggiornamento di voci di accesso

È possibile aggiornare una voce di accesso utilizzando AWS Management Console o il AWS CLI.

AWS Management Console

Per aggiornare una voce di accesso

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli il nome del cluster in cui desideri creare una voce di accesso.
3. Scegli la scheda Accesso.
4. Scegli la voce di accesso che desideri aggiornare.
5. Scegli Modifica.
6. Per Nome utente, puoi modificare il valore esistente.
7. Per Gruppi, puoi rimuovere i nomi dei gruppi esistenti o aggiungere nuovi nomi dei gruppi. Se esistono i seguenti nomi di gruppi, non rimuoverli: system:nodes o system:bootstrappers. La rimozione di questi gruppi può causare un malfunzionamento del cluster. Se non specifichi nomi dei gruppi e desideri utilizzare l'autorizzazione Amazon EKS, associa una policy di accesso in un passaggio successivo.
8. Per Tag, puoi assegnare etichette alla voce di accesso. Ad esempio, per facilitare la ricerca di tutte le risorse con lo stesso tag. Puoi anche rimuovere i tag esistenti.
9. Seleziona Salvataggio delle modifiche.
10. Se desideri associare una policy di accesso alla voce, consulta la sezione Associazione e dissociazione delle policy di accesso alle/dalle voci di accesso.

AWS CLI

Prerequisito

Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella tua home directory nella Guida AWS CLI per l'AWS CloudShell utente.

Per aggiornare una voce di accesso

Sostituisci *`my-cluster`* con il nome del cluster, *`111122223333`* con il tuo Account AWS ID e *`EKS-My-Cluster-My-Namespace-Viewers`* con il nome di un ruolo IAM.

```
aws eks update-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers --kubernetes-
groups Viewers
```

Non è possibile utilizzare l'opzione `--kubernetes-groups` se il tipo di voce di accesso è un valore diverso da `STANDARD`. Inoltre, non è possibile associare una policy di accesso a una voce di accesso con un tipo diverso da `STANDARD`.

Eliminazione di voci di accesso

Se scopri di aver eliminato una voce di accesso per errore, puoi sempre ricrearla. Se la voce di accesso che stai eliminando è associata a policy di accesso, le associazioni vengono eliminate automaticamente. Non è necessario dissociare le policy di accesso da una voce di accesso prima di eliminare quest'ultima.

È possibile eliminare AWS Management Console AWS CLI una voce di accesso utilizzando o il.

AWS Management Console

Per eliminare una voce di accesso

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli il nome del cluster da cui desideri eliminare una voce di accesso.
3. Scegli la scheda Accesso.
4. Nell'elenco Voci di accesso, seleziona la voce di accesso da eliminare.
5. Scegli Elimina.
6. Nella finestra di dialogo di conferma, seleziona Elimina.

AWS CLI

Prerequisito

Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella tua home directory nella Guida AWS CLI per l'AWS CloudShell utente.

Per eliminare una voce di accesso

Sostituisci *`my-cluster`* con il nome del tuo cluster, *`111122223333`* con il tuo Account AWS ID e *`my-role`* con il nome del ruolo IAM a cui non desideri più avere accesso al tuo cluster.

```
aws eks delete-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role
```

Associazione e dissociazione delle policy di accesso alle/dalle voci di accesso

È possibile assegnare una o più policy di accesso alle voci di accesso di tipo `STANDARD`. Amazon EKS concede automaticamente agli altri tipi di voci di accesso le autorizzazioni necessarie per funzionare correttamente nel cluster. Le policy di accesso di Amazon EKS includono le autorizzazioni di Kubernetes, non le autorizzazioni IAM. Prima di associare una policy di accesso a una voce di accesso, assicurati di conoscere le autorizzazioni di Kubernetes incluse in ogni policy di accesso. Per ulteriori informazioni, consulta Autorizzazioni della policy di accesso. Se nessuna delle policy di accesso soddisfa i tuoi requisiti, non associarne alcuna a una voce di accesso. Specifica invece uno o più nomi dei gruppi per la voce di accesso e crea/gestisci oggetti di controllo degli accessi basato sui ruoli di Kubernetes. Per ulteriori informazioni, consulta Creazione di voci di accesso.

Prerequisiti

- Una voce di accesso esistente. Per crearne uno, consulta Creazione di voci di accesso.
- Un AWS Identity and Access Management ruolo o un utente con le seguenti autorizzazioni:`ListAccessEntries`,`DescribeAccessEntry`,`UpdateAccessEntry`, `ListAccessPoliciesAssociateAccessPolicy`, `eDisassociateAccessPolicy`. Per ulteriori informazioni, consulta Actions defined by Amazon Elastic Kubernetes Service nella Documentazione di riferimento per l'autorizzazione ai servizi.

Prima di associare policy di accesso a voci di accesso, considera i seguenti requisiti:

- È possibile associare più policy di accesso a ciascuna voce di accesso, ma è possibile associare ogni policy a una voce di accesso solo una volta. Se si associano più policy di accesso, il principale IAM della voce di accesso dispone di tutte le autorizzazioni incluse in tutte le policy di accesso associate.
- È possibile applicare una policy di accesso a tutte le risorse di un cluster o specificando il nome di uno o più spazi dei nomi di Kubernetes. È possibile utilizzare caratteri jolly per il nome di un spazio dei nomi. Ad esempio, se si desidera applicare una policy di accesso a tutti gli spazi dei nomi che iniziano con`dev-`, è possibile specificare `dev-*` come nome dello spazio dei nomi. È necessario assicurarsi che gli spazi dei nomi siano presenti sul cluster e che quanto digitato corrisponda esattamente al nome reale dello spazio dei nomi sul cluster. Amazon EKS, infatti, non conferma l'ortografia o l'esistenza degli spazi dei nomi sul cluster.

- È possibile modificare l'ambito di accesso per una policy di accesso dopo averla associata a una voce di accesso. Se hai limitato l'ambito della policy di accesso agli spazi dei nomi di Kubernetes, puoi aggiungere e rimuovere gli spazi dei nomi per l'associazione, secondo necessità.
- Se associ una policy di accesso a una voce di accesso in cui sono specificati anche i nomi dei gruppi, il principale IAM dispone di tutte le autorizzazioni in tutte le policy di accesso associate. Inoltre, dispone di tutte le autorizzazioni in qualsiasi oggetto `Role` o `ClusterRole` di Kubernetes specificato in qualsiasi oggetto `RoleBinding` e Kubernetes `Role` che specifica i nomi dei gruppi.
- Se esegui il comando `kubectl auth can-i --list`, non vedrai alcun permesso di Kubernetes assegnato dalle policy di accesso associate a una voce di accesso per il principale IAM che stai utilizzando quando esegui il comando. Il comando mostra le autorizzazioni di Kubernetes solo se le hai concesse a oggetti `Role` o `ClusterRole` di Kubernetes che hai associato ai nomi dei gruppi o al nome utente che hai specificato per una voce di accesso.
- Se impersoni un utente o un gruppo di Kubernetes quando interagisci con oggetti Kubernetes sul cluster, ad esempio utilizzando il comando `kubectl` con **`--as`** ***`username`*** o **`--as-group`** ***`group-name`***, stai forzando l'uso dell'autorizzazione RBAC di Kubernetes. Di conseguenza, il principale IAM non dispone di autorizzazioni assegnate da alcuna policy di accesso associata alla voce di accesso. Le uniche autorizzazioni Kubernetes di cui dispone l'utente o il gruppo che il principale IAM sta impersonando sono le autorizzazioni Kubernetes che gli hai concesso negli oggetti `Role` o `ClusterRole` di Kubernetes che hai associato ai nomi dei gruppi o al nome utente. Per garantire che il tuo principale IAM abbia le autorizzazioni nelle policy di accesso associate, evita di impersonare un utente o un gruppo di Kubernetes. Il principale IAM avrà comunque anche tutte le autorizzazioni che gli hai concesso negli oggetti `Role` o `ClusterRole` di Kubernetes che hai associato ai nomi dei gruppi o al nome utente che hai specificato per la voce di accesso. Per ulteriori informazioni, consulta la sezione User impersonation nella documentazione di Kubernetes.

È possibile associare un criterio di accesso a una voce di accesso utilizzando AWS Management Console o il AWS CLI.

AWS Management Console

Per associare una policy di accesso a una voce di accesso utilizzando la AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli il nome del cluster con una voce di accesso da associare a una policy di accesso.

3. Scegli la scheda Accesso.
4. Se il tipo di voce di accesso è Standard, puoi associare o dissociare le policy di accesso di Amazon EKS. Se il tipo di voce di accesso è diverso da Standard, questa opzione non è disponibile.
5. Scegli Associa policy di accesso.
6. Per Nome della policy, seleziona la policy con le autorizzazioni che desideri assegnare al principale IAM. Per visualizzare i permessi inclusi in ciascuna policy, consulta Autorizzazioni della policy di accesso.
7. Per Ambito di accesso, scegli un ambito di accesso. Se scegli Cluster, le autorizzazioni nella policy di accesso vengono concesse al principale IAM per le risorse in tutti gli spazi dei nomi di Kubernetes. Se scegli Spazio dei nomi Kubernetes, puoi scegliere Aggiungi nuovo spazio dei nomi. Nel campo Spazio dei nomi che appare, puoi inserire il nome di uno spazio dei nomi Kubernetes sul tuo cluster. Se desideri che il principale IAM disponga delle autorizzazioni su più spazi dei nomi, puoi inserirne più di uno.
8. Scegli Aggiungi policy di accesso.

AWS CLI

Prerequisito

Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.

Per associare una policy di accesso a una voce di accesso

1. Visualizza le policy di accesso disponibili.

```
aws eks list-access-policies --output table
```

Di seguito viene riportato un output di esempio:

```
-------------------------------------------------------------------------------------
|                                        ListAccessPolicies
                      |
+-------------------------------------------------------------------------------------
+
||                                          accessPolicies
                     ||
|+-------------------------------------------------------------------
+-------------------------------+|
||                                arn                                |
     name              ||
|+-------------------------------------------------------------------
+-------------------------------+|
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy        |
 AmazonEKSAdminPolicy         ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy |
 AmazonEKSClusterAdminPolicy  ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy         |
 AmazonEKSEditPolicy          ||
||  arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy         |
 AmazonEKSViewPolicy          ||
|+-------------------------------------------------------------------
+-------------------------------+|
```

Per visualizzare i permessi inclusi in ciascuna policy, consulta Autorizzazioni della policy di accesso.

2. Visualizza le tue voci di accesso esistenti. Sostituisci *my-cluster* con il nome del tuo cluster.

```
aws eks list-access-entries --cluster-name my-cluster
```

Di seguito viene riportato un output di esempio:

```
{
    "accessEntries": [
        "arn:aws:iam::111122223333:role/my-role",
        "arn:aws:iam::111122223333:user/my-user"
    ]
```

```
}
```

3. Associa una policy di accesso a una voce di accesso. L'esempio seguente associa la policy di accesso `AmazonEKSViewPolicy` a una voce di accesso. Ogni volta che il ruolo IAM *my-role* tenta di accedere agli oggetti Kubernetes sul cluster, Amazon EKS autorizzerà il ruolo a utilizzare le autorizzazioni della policy per accedere agli oggetti Kubernetes solo negli spazi dei nomi *my-namespace1* e *my-namespace2* di Kubernetes. Sostituisci *my-cluster* con il nome del tuo cluster, *111122223333* con il tuo ID Account AWS e *my-role* con il nome del ruolo IAM che desideri che Amazon EKS autorizzi ad accedere agli oggetti del cluster Kubernetes.

```
aws eks associate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --access-scope type=namespace,namespaces=my-namespace1,my-namespace2 --
policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy
```

Se desideri che il principale IAM disponga delle autorizzazioni a livello di cluster, sostituisci **type=namespace,namespaces=*my-namespace1*,*my-namespace2*** con **type=cluster**. Se desideri associare più policy di accesso alla voce di accesso, esegui il comando più volte, ogni volta con una policy di accesso unica. Ogni policy di accesso associata ha il proprio ambito.

> **Note**
>
> Se in seguito desideri modificare l'ambito di una policy di accesso associata, esegui nuovamente il comando precedente con il nuovo ambito. Ad esempio, se desideri rimuovere *my-namespace2*, esegui nuovamente il comando utilizzando solo **type=namespace,namespaces=*my-namespace1***. Se desideri modificare l'ambito da **namespace** a **cluster**, esegui nuovamente il comando utilizzando **type=cluster** e rimuovendo **type=namespace,namespaces=*my-namespace1*,*my-namespace2***.

Per dissociare una policy di accesso da una voce di accesso

1. Determina quali policy di accesso sono associate a una voce di accesso.

```
aws eks list-associated-access-policies --cluster-name my-cluster --principal-
arn arn:aws:iam::111122223333:role/my-role
```

Di seguito viene riportato un output di esempio:

```
{
    "clusterName": "my-cluster",
    "principalArn": "arn:aws:iam::111122223333",
    "associatedAccessPolicies": [
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-
policy/AmazonEKSViewPolicy",
            "accessScope": {
                "type": "cluster",
                "namespaces": []
            },
            "associatedAt": "2023-04-17T15:25:21.675000-04:00",
            "modifiedAt": "2023-04-17T15:25:21.675000-04:00"
        },
        {
            "policyArn": "arn:aws:eks::aws:cluster-access-
policy/AmazonEKSAdminPolicy",
            "accessScope": {
                "type": "namespace",
                "namespaces": [
                    "my-namespace1",
                    "my-namespace2"
                ]
            },
            "associatedAt": "2023-04-17T15:02:06.511000-04:00",
            "modifiedAt": "2023-04-17T15:02:06.511000-04:00"
        }
    ]
}
```

Nell'esempio precedente, il principale IAM per questa voce di accesso ha permessi di visualizzazione in tutti gli spazi dei nomi del cluster e permessi di amministratore per due spazi dei nomi Kubernetes.

2. Dissociazione di una policy di accesso da una voce di accesso. In questo esempio, la policy `AmazonEKSAdminPolicy` è dissociata da una voce di accesso. Tuttavia, il principale IAM

mantiene le autorizzazioni nella policy di accesso di `AmazonEKSViewPolicy` per gli oggetti negli spazi dei nomi *`my-namespace1`* e *`my-namespace2`*, poiché tale policy di accesso non è dissociata dalla voce di accesso.

```
aws eks disassociate-access-policy --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/my-role \
    --policy-arn arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy
```

Autorizzazioni della policy di accesso

Le policy di accesso includono `rules` che contengono `verbs` (autorizzazioni) e `resources` di Kubernetes. Le policy di accesso non includono le autorizzazioni o le risorse IAM. Analogamente agli oggetti `Role` e `ClusterRole` di Kubernetes, le policy di accesso includono solo `allow` `rules`. Non è possibile modificare il contenuto di una policy di accesso. Non è possibile creare policy di accesso personalizzate. Se le autorizzazioni nelle policy di accesso non soddisfano le tue esigenze, crea oggetti RBAC di Kubernetes e specifica nomi dei gruppi per le tue voci di accesso. Per ulteriori informazioni, consulta Creazione di voci di accesso. Le autorizzazioni contenute nelle policy di accesso sono simili alle autorizzazioni nei ruoli del cluster rivolti agli utenti di Kubernetes. Per ulteriori informazioni, consulta User-facing roles nella documentazione di Kubernetes.

Scegli una policy di accesso per visualizzarne il contenuto. Ogni riga di ciascuna tabella in ogni policy di accesso rappresenta una regola distinta.

AmazonEks AdminPolicy

Questa policy di accesso include autorizzazioni che concedono a un principale IAM la maggior parte delle autorizzazioni per le risorse. Quando è associata a una voce di accesso, il relativo ambito di accesso è in genere uno o più spazio dei nomi Kubernetes. Se invece desideri che un principale IAM disponga dell'accesso di amministratore a tutte le risorse del cluster, allora associa la policy di accesso AmazonEks ClusterAdminPolicy alla tua voce di accesso.

ARN: `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminPolicy`

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizzazioni)
`apps`	`daemonsets` , `deployments` , `deployments/`	`create`, `delete`, `deletecollection` , `patch`, `update`

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizz azioni)
	rollback , deploymen ts/scale , replicase ts , replicasets/ scale, statefulsets , statefulsets/scale	
apps	controllerrevisions , daemonsets , daemonset s/status , deploymen ts , deployments/scale , deployments/status , replicasets , replicase ts/scale , replicase ts/status , statefuls ets , statefulsets/ scale , statefulsets/ status	get, list, watch
authorization.k8s.io	localsubjectaccess reviews	create
autoscaling	horizontalpodautos calers	create, delete, deletecol lection , patch, update
autoscaling	horizontalpodautos calers , horizonta lpodautoscalers/st atus	get, list, watch
batch	cronjobs, jobs	create, delete, deletecol lection , patch, update
batch	cronjobs, cronjobs/ status , jobs, jobs/stat us	get, list, watch

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizz azioni)
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , deploymen ts , deployments/ rollback , deploymen ts/scale , ingresses , networkpolicies , replicasets , replicase ts/scale , replicati oncontrollers/scale	create, delete, deletecol lection , patch, update
extensions	daemonsets , daemonset s/status , deploymen ts , deployments/scale , deployments/status , ingresses , ingresses /status , networkpo licies , replicasets , replicasets/scale , replicasets/status , replicationcontrol lers/scale	get, list, watch
networking.k8s.io	ingresses , ingresses /status , networkpo licies	get, list, watch
networking.k8s.io	ingresses , networkpo licies	create, delete, deletecol lection , patch, update
policy	poddisruptionbudgets	create, delete, deletecol lection , patch, update

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizz azioni)
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch
rbac.authorization.k8s.io	rolebindings , roles	create, delete, deletecollection , get, list, patch, update, watch
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get,list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	configmaps , events, persistentvolumeclaims , replicationcontrollers , replicationcontrollers/scale , secrets, serviceaccounts , services, services/proxy	create, delete, deletecollection , patch, update

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizz azioni)
	pods, pods/attach , pods/exec , pods/port forward , pods/proxy	create, delete, deletecol lection , patch, update
	serviceaccounts	impersonate
	bindings, events, limitranges , namespace s/status , pods/log, pods/status , replicati oncontrollers/stat us , resourcequotas , resourcequotas/sta tus	get, list, watch
	namespaces	get,list, watch

AmazonEks ClusterAdminPolicy

Questa policy di accesso include le autorizzazioni che concedono a un principale IAM l'accesso di amministratore per un cluster. Se associata a una voce di accesso, il relativo ambito di accesso è in genere il cluster, anziché uno spazio dei nomi Kubernetes. Se desideri che un principale IAM abbia un ambito amministrativo più limitato, valuta invece la possibilità di associare la policy di accesso AmazonEks AdminPolicy alla tua voce di accesso.

ARN: arn:aws:eks::aws:cluster-access-policy/AmazonEKSClusterAdminPolicy

Gruppi di API Kubernetes	Kubernetes nonResourceURLs	Risorse Kubernetes	Verbi Kubernetes (autorizzazioni)
*		*	*
	*		*

AmazonEks AdminViewPolicy

Questa politica di accesso include autorizzazioni che garantiscono a un IAM l'accesso principale per elencare/visualizzare tutte le risorse in un cluster. Nota che questo include Secrets. Kubernetes

ARN: `arn:aws:eks::aws:cluster-access-policy/AmazonEKSAdminViewPolicy`

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizz azioni)
*	*	get, list, watch

AmazonEks EditPolicy

Questa policy di accesso include autorizzazioni che consentono a un principale IAM di modificare la maggior parte delle risorse Kubernetes.

ARN: `arn:aws:eks::aws:cluster-access-policy/AmazonEKSEditPolicy`

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizz azioni)
apps	daemonsets , deployments , deployments/rollback , deployments/scale , replicasets , replicasets/scale, statefulsets , statefulsets/scale	create, delete, deletecollection , patch, update
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefuls	get, list, watch

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizzazioni)
	ets , statefulsets/scale , statefulsets/status	
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
autoscaling	horizontalpodautoscalers	create, delete, deletecollection , patch, update
batch	cronjobs, jobs	create, delete, deletecollection , patch, update
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , deployments , deployments/rollback , deployments/scale , ingresses , networkpolicies , replicasets , replicasets/scale , replicationcontrollers/scale	create, delete, deletecollection , patch, update

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizz azioni)
extensions	daemonsets , daemonset s/status , deploymen ts , deployments/scale , deployments/status , ingresses , ingresses /status , networkpo licies , replicasets , replicasets/scale , replicasets/status , replicationcontrol lers/scale	get, list, watch
networking.k8s.io	ingresses , networkpo licies	create, delete, deletecol lection , patch, update
networking.k8s.io	ingresses , ingresses /status , networkpo licies	get, list, watch
policy	poddisruptionbudgets	create, delete, deletecol lection , patch, update
policy	poddisruptionbudgets , poddisruptionbudge ts/status	get, list, watch
	namespaces	get, list, watch
	pods/attach , pods/exec , pods/portforward , pods/proxy , secrets, services/proxy	get, list, watch
	serviceaccounts	impersonate

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizz azioni)
	pods, pods/attach , pods/exec , pods/port forward , pods/proxy	create, delete, deletecol lection , patch, update
	configmaps , events, persistentvolumecl aims , replicati oncontrollers , replicationcontrol lers/scale , secrets, serviceaccounts , services, services/ proxy	create, delete, deletecol lection , patch, update
	configmaps , endpoints , persistentvolumecl aims , persisten tvolumeclaims/stat us , pods, replicati oncontrollers , replicationcontrol lers/scale , serviceac counts , services, services/status	get, list, watch
	bindings, events, limitranges , namespace s/status , pods/log, pods/status , replicati oncontrollers/stat us , resourcequotas , resourcequotas/sta tus	get, list, watch

AmazonEks ViewPolicy

Questa policy di accesso include autorizzazioni che consentono a un principale IAM di visualizzare la maggior parte delle risorse Kubernetes.

ARN: `arn:aws:eks::aws:cluster-access-policy/AmazonEKSViewPolicy`

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizzazioni)
apps	controllerrevisions , daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , replicasets , replicasets/scale , replicasets/status , statefulsets , statefulsets/scale , statefulsets/status	get, list, watch
autoscaling	horizontalpodautoscalers , horizontalpodautoscalers/status	get, list, watch
batch	cronjobs, cronjobs/status , jobs, jobs/status	get, list, watch
discovery.k8s.io	endpointslices	get, list, watch
extensions	daemonsets , daemonsets/status , deployments , deployments/scale , deployments/status , ingresses , ingresses/status , networkpo	get, list, watch

Gruppi di API Kubernetes	Risorse Kubernetes	Verbi Kubernetes (autorizzazioni)
	licies , replicasets , replicasets/scale , replicasets/status , replicationcontrollers/scale	
networking.k8s.io	ingresses , ingresses/status , networkpolicies	get, list, watch
policy	poddisruptionbudgets , poddisruptionbudgets/status	get, list, watch
	configmaps , endpoints , persistentvolumeclaims , persistentvolumeclaims/status , pods, replicationcontrollers , replicationcontrollers/scale , serviceaccounts , services, services/status	get, list, watch
	bindings, events, limitranges , namespaces/status , pods/log, pods/status , replicationcontrollers/status , resourcequotas , resourcequotas/status	get, list, watch
	namespaces	get, list, watch

Aggiornamenti della policy di accesso

Visualizza i dettagli sugli aggiornamenti alle policy di accesso, da quando sono stati introdotti. Per gli avvisi automatici sulle modifiche apportate a questa pagina, effettua l'abbonamento al feed RSS nella pagina della cronologia dei documenti di Amazon EKS.

Modifica	Descrizione	Data
Add `AmazonEKS AdminView Policy`	Aggiungi una nuova politica per un accesso esteso alle visualizzazioni, incluse risorse come Secrets.	23 aprile 2024
Policy di accesso introdotte.	Amazon EKS ha introdotto policy di accesso.	29 maggio 2023

Migrazione delle voci **`aws-auth ConfigMap`** esistenti alle voci di accesso

Se hai aggiunto voci a `aws-auth ConfigMap` sul tuo cluster, ti consigliamo di creare voci di accesso per le voci esistenti in `aws-auth ConfigMap`. Dopo aver creato le voci di accesso, puoi rimuovere le voci da `ConfigMap`. Non è possibile associare le policy di accesso alle voci presenti in `aws-auth ConfigMap`. Se desideri associare policy di accesso ai tuoi principali IAM, crea voci di accesso.

> **Important**
>
> Non rimuovere le voci `aws-auth ConfigMap` esistenti create da Amazon EKS quando hai aggiunto un gruppo di nodi gestiti o un profilo Fargate al tuo cluster. Se rimuovi le voci che Amazon EKS ha creato in `ConfigMap`, il cluster non funzionerà correttamente. Tuttavia, puoi rimuovere qualsiasi voce per i gruppi di nodi autogestiti dopo aver creato le voci di accesso per tali gruppi.

Prerequisiti

- Familiarità con le voci di accesso e le policy di accesso. Per ulteriori informazioni, consulta Gestire le voci di accesso e Associazione e dissociazione delle policy di accesso alle/dalle voci di accesso.

- Un cluster esistente con una versione della piattaforma corrispondente o successiva alle versioni elencate nei Prerequisiti dell'argomento Consentire ai ruoli o agli utenti IAM di accedere agli oggetti Kubernetes sul cluster Amazon EKS.
- La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.
- Le autorizzazioni di Kubernetes per modificare `aws-auth ConfigMap` nello spazio dei nomi `kube-system`.
- Un AWS Identity and Access Management ruolo o un utente con le seguenti autorizzazioni: `CreateAccessEntry` e. `ListAccessEntries` Per ulteriori informazioni, consulta Actions defined by Amazon Elastic Kubernetes Service nella Documentazione di riferimento per l'autorizzazione ai servizi.

Per migrare una voce dalla tua **`aws-auth ConfigMap`** a una voce di accesso

1. Verifica le voci esistenti in `aws-auth ConfigMap`. Sostituisci *`my-cluster`* con il nome del tuo cluster.

   ```
   eksctl get iamidentitymapping --cluster my-cluster
   ```

   Di seguito viene riportato un output di esempio:

   ```
   ARN
                USERNAME                                GROUPS
                            ACCOUNT
   arn:aws:iam::111122223333:role/EKS-my-cluster-Admins
                Admins                                  system:masters
   arn:aws:iam::111122223333:role/EKS-my-cluster-my-namespace-Viewers
                my-namespace-Viewers                    Viewers
   arn:aws:iam::111122223333:role/EKS-my-cluster-self-managed-ng-1
                               system:node:{{EC2PrivateDNSName}}
    system:bootstrappers,system:nodes
   arn:aws:iam::111122223333:user/my-user
                my-user
   arn:aws:iam::111122223333:role/EKS-my-cluster-fargateprofile1
                               system:node:{{SessionName}}
    system:bootstrappers,system:nodes,system:node-proxier
   ```

```
arn:aws:iam::111122223333:role/EKS-my-cluster-managed-ng
                              system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

2. Crea voci di accesso per tutte le voci `ConfigMap` che hai creato e che sono state restituite nell'output precedente. Quando crei le voci di accesso, assicurati di specificare gli stessi valori per `ARN`, `USERNAME`, `GROUPS` e `ACCOUNT` restituiti nell'output. Nell'output di esempio, dovresti creare voci di accesso per tutte le voci tranne le ultime due, poiché tali voci sono state create da Amazon EKS per un profilo Fargate e un gruppo di nodi gestito.
3. Elimina le voci da `ConfigMap` per tutte le voci di accesso che hai creato. Se non elimini la voce da `ConfigMap`, le impostazioni per la voce di accesso per l'ARN principale IAM hanno la precedenza sulla voce `ConfigMap`. Sostituisci *111122223333* con il tuo Account AWS ID e *EKS-MY-Cluster-My-Namespace-Viewers con il nome del ruolo* nella voce del tuo. `ConfigMap` Se la voce che stai rimuovendo è per un utente IAM, anziché per un ruolo IAM, sostituisci **`role`** con **`user`** e *EKS-my-cluster-my-namespace-Viewers* con il nome utente.

```
eksctl delete iamidentitymapping --arn arn:aws:iam::111122223333:role/EKS-my-
cluster-my-namespace-Viewers --cluster my-cluster
```

Concedere al principale IAM l'accesso al cluster

> **Important**
>
> `aws-auth`ConfigMapÈ obsoleto. Il metodo consigliato per gestire l'accesso alle API è Access Entries. Kubernetes

L'accesso al cluster utilizzando i principali (IAM) è abilitato dall'autenticatore AWS IAM per Kubernetes, che viene eseguito sul piano di controllo (control-plane) di Amazon EKS. L'autenticatore riceve le informazioni di configurazione da `aws-auth` `ConfigMap`. Per tutte le impostazioni di `ConfigMap` `aws-auth`, consulta Formato di configurazione completo su GitHub.

Aggiunta di principali IAM al cluster Amazon EKS

Quando si crea un cluster Amazon EKS, il principale IAM che crea il cluster riceve automaticamente le autorizzazioni `system:masters` nella configurazione del controllo degli accessi basato sul ruolo (RBAC) nel piano di controllo (control-plane) di Amazon EKS. Questo principale IAM non

viene visualizzato in una configurazione visibile qualsiasi, quindi assicurati di tenere traccia di quale principale IAM ha originariamente creato il cluster. Per concedere a ulteriori principali IAM la capacità di interagire con il cluster, devi modificare `aws-auth ConfigMap` all'interno di Kubernetes e creare un Kubernetes `rolebinding` o `clusterrolebinding` con il nome di un `group` specificato in `aws-auth ConfigMap`.

> **Note**
>
> Per ulteriori informazioni sulla configurazione del controllo degli accessi basato sul ruolo (RBAC) di Kubernetes, consulta Utilizzo dell'autorizzazione RBAC nella documentazione di Kubernetes.

Per aggiungere principali IAM a un cluster Amazon EKS

1. Determina quali credenziali `kubectl` vengono utilizzate per accedere al cluster. Sul computer, è possibile vedere quali credenziali `kubectl` utilizza con il seguente comando. Sostituisci ***~/.kube/config*** con il percorso del file `kubeconfig` se non si utilizza il percorso predefinito.

   ```
   cat ~/.kube/config
   ```

   Di seguito viene riportato un output di esempio.

   ```
   [...]
   contexts:
   - context:
       cluster: my-cluster.region-code.eksctl.io
       user: admin@my-cluster.region-code.eksctl.io
     name: admin@my-cluster.region-code.eksctl.io
   current-context: admin@my-cluster.region-code.eksctl.io
   [...]
   ```

   Nell'output dell'esempio precedente, le credenziali per un utente denominato *`admin`* sono configurate per un cluster denominato *`my-cluster`*. Se si tratta dell'utente che ha creato il cluster, ha già accesso al cluster. Se l'utente non ha creato il cluster, è necessario completare i passaggi rimanenti per abilitare l'accesso al cluster per gli altri principali IAM. Le best practice IAM consigliano di concedere le autorizzazioni ai ruoli anziché agli utenti. Per visualizzare gli altri principali che hanno attualmente accesso al cluster, esegui il comando seguente:

```
kubectl describe -n kube-system configmap/aws-auth
```

Di seguito viene riportato un output di esempio.

```
Name:         aws-auth
Namespace:    kube-system
Labels:       <none>
Annotations:  <none>

Data
====
mapRoles:
----
- groups:
  - system:bootstrappers
  - system:nodes
  rolearn: arn:aws:iam::111122223333:role/my-node-role
  username: system:node:{{EC2PrivateDNSName}}


BinaryData
====

Events:  <none>
```

L'esempio precedente è una `aws-auth` `ConfigMap` predefinita. Solo il ruolo dell'istanza del nodo ha accesso al cluster.

2. Assicurati di disporre di `roles` e `rolebindings` o `clusterroles` e `clusterrolebindings` di Kubernetes da mappare ai principali IAM. Per ulteriori informazioni su queste risorse, consulta Utilizzo dell'autorizzazione RBAC nella documentazione di Kubernetes.

   1. Visualizza i `roles` o `clusterroles` Kubernetes esistenti. L'ambito di `Roles` è `namespace` mentre l'ambito di `clusterroles` è il cluster.

      ```
      kubectl get roles -A
      ```

      ```
      kubectl get clusterroles
      ```

2. Visualizza i dettagli dei valori `role` o `clusterrole` restituiti nell'output precedente e verifica di disporre delle autorizzazioni (`rules`) da concedere ai principali IAM del cluster.

   Sostituisci *`role-name`* con un nome del `role` restituito nell'output del comando precedente. Sostituisci *`kube-system`* con lo spazio dei nomi del `role`.

   ```
   kubectl describe role role-name -n kube-system
   ```

   Sostituisci *`cluster-role-name`* con un nome del `clusterrole` restituito nell'output del comando precedente.

   ```
   kubectl describe clusterrole cluster-role-name
   ```

3. Visualizza i `rolebindings` o `clusterrolebindings` Kubernetes esistenti. L'ambito di `Rolebindings` è `namespace` mentre l'ambito di `clusterrolebindings` è il cluster.

   ```
   kubectl get rolebindings -A
   ```

   ```
   kubectl get clusterrolebindings
   ```

4. Visualizza i dettagli di `rolebinding` o `clusterrolebinding` e verifica che dispongano di un `role` or `clusterrole` della fase precedente elencato come un `roleRef` e un nome di gruppo elencato per `subjects`.

   Sostituisci *`role-binding-name`* con un nome del `rolebinding` restituito nell'output del comando precedente. Sostituisci *`kube-system`* con lo `namespace` del `rolebinding`.

   ```
   kubectl describe rolebinding role-binding-name -n kube-system
   ```

   Di seguito viene riportato un output di esempio.

   ```
   apiVersion: rbac.authorization.k8s.io/v1
   kind: RoleBinding
   metadata:
     name: eks-console-dashboard-restricted-access-role-binding
     namespace: default
   subjects:
   - kind: Group
     name: eks-console-dashboard-restricted-access-group
   ```

```
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: Role
  name: eks-console-dashboard-restricted-access-role
  apiGroup: rbac.authorization.k8s.io
```

Sostituisci *cluster-role-binding-name* con un nome del `clusterrolebinding` restituito nell'output del comando precedente.

```
kubectl describe clusterrolebinding cluster-role-binding-name
```

Di seguito viene riportato un output di esempio.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks-console-dashboard-full-access-binding
subjects:
- kind: Group
  name: eks-console-dashboard-full-access-group
  apiGroup: rbac.authorization.k8s.io
roleRef:
  kind: ClusterRole
  name: eks-console-dashboard-full-access-clusterrole
  apiGroup: rbac.authorization.k8s.io
```

3. Modificare `aws-auth ConfigMap`. Per aggiornare `ConfigMap` puoi usare uno strumento adeguato, ad esempio `eksctl`, oppure puoi eseguire l'aggiornamento in modo manuale tramite modifica.

> **Important**
>
> Ti consigliamo di utilizzare `eksctl`, o uno strumento simile, per modificare `ConfigMap`. Per informazioni su altri strumenti che è possibile utilizzare, consulta Utilizzo degli strumenti per apportare modifiche alla `aws-authConfigMap` nelle guide alle best practice di Amazon EKS. Una formattazione impropria di `aws-auth ConfigMap` può causare la perdita dell'accesso al cluster.

eksctl

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

1. Visualizza le mappature correnti in `ConfigMap`. Sostituisci *`my-cluster`* con il nome del cluster. *`region-code`*Sostituiscilo con Regione AWS quello in cui si trova il tuo cluster.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

   Di seguito viene riportato un output di esempio:

```
ARN
                  USERNAME                                GROUPS
            ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

2. Aggiungi una mappatura per un ruolo. Sostituisci *`my-role`* con il nome del tuo ruolo. Sostituisci *`eks-console-dashboard-full-access-group`* con il nome del gruppo specificato nell'oggetto Kubernetes `RoleBinding` o `ClusterRoleBinding`. Sostituisci *`111122223333`* con l'ID del tuo account. Puoi sostituire *`admin`* con un nome a tua scelta.

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws:iam::111122223333:role/my-role --username admin --group eks-
console-dashboard-full-access-group \
    --no-duplicate-arns
```

> **Important**
>
> L'ARN del ruolo non può includere un percorso, ad esempio `role/my-team/developers/my-role`. Il formato dell'ARN deve essere

`arn:aws:iam::111122223333:role/my-role`. In questo esempio, `my-team/developers/` deve essere rimosso.

Di seguito viene riportato un output di esempio.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-
role" to auth ConfigMap
```

3. Aggiungi una mappatura per un utente. Le best practice IAM consigliano di concedere le autorizzazioni ai ruoli anziché agli utenti. Sostituisci *my-user* con il tuo nome utente. Sostituisci *eks-console-dashboard-restricted-access-group* con il nome del gruppo specificato nell'oggetto Kubernetes `RoleBinding` o `ClusterRoleBinding`. Sostituisci *111122223333* con l'ID del tuo account. Puoi sostituire *my-user* con un nome a tua scelta.

```
eksctl create iamidentitymapping --cluster my-cluster --region=region-code \
    --arn arn:aws:iam::111122223333:user/my-user --username my-user --
group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

Di seguito viene riportato un output di esempio.

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-
user" to auth ConfigMap
```

4. Visualizza nuovamente le mappature nella `ConfigMap`.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

Di seguito viene riportato un output di esempio.

```
ARN
                    USERNAME                                GROUPS
                      ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
```

```
arn:aws:iam::111122223333:role/admin
                        my-role                                 eks-console-
dashboard-full-access-group
arn:aws:iam::111122223333:user/my-user
                        my-user                                 eks-console-
dashboard-restricted-access-group
```

Edit ConfigMap manually

1. Aprire la `ConfigMap` per la modifica.

```
kubectl edit -n kube-system configmap/aws-auth
```

> **Note**
>
> Se si verifica un errore di tipo "`Error from server (NotFound): configmaps "aws-auth" not found`", usa la procedura descritta in Applica la `aws-auth` `ConfigMap` al cluster per applicare lo stock `ConfigMap`.

2. Aggiungi i tuoi principali IAM alla `ConfigMap`. Un gruppo IAM non è un principale IAM, quindi non può essere aggiunto alla `ConfigMap`.
   - Per aggiungere un ruolo IAM (ad esempio, per utenti federati): aggiungere i dettagli del ruolo alla sezione `mapRoles` della `ConfigMap`, in `data`. Aggiungi questa sezione se non esiste già nel file. Ogni voce supporta i seguenti parametri:
     - rolearn: l'ARN; del ruolo IAM da aggiungere. Questo valore non può includere un percorso. Ad esempio, non puoi specificare un ARN, come `arn:aws:iam::111122223333:role/my-team/developers/role-name`. L'ARN deve essere `arn:aws:iam::111122223333:role/role-name`.
     - username: il nome utente all'interno di Kubernetes da associare al ruolo IAM.
     - groups: il gruppo o l'elenco dei gruppi Kubernetes a cui mappare il ruolo. Tale gruppo può essere un gruppo predefinito o un gruppo specificato in un `clusterrolebinding` o un `rolebinding`. Per ulteriori informazioni consulta Ruoli predefiniti e associazioni di ruoli nella documentazione di Kubernetes.
   - Per aggiungere un utente IAM: le best practice IAM consigliano di concedere le autorizzazioni ai ruoli anziché agli utenti. Aggiungi i dettagli dell'utente alla sezione

`mapUsers` della `ConfigMap`, in `data`. Aggiungi questa sezione se non esiste già nel file. Ogni voce supporta i seguenti parametri:

- userarn: L'ARN; dell'utente IAM da aggiungere.
- username: il nome utente all'interno di Kubernetes da associare all'utente IAM.
- groups: il gruppo o l'elenco dei gruppi Kubernetes a cui mappare l'utente. Tale gruppo può essere un gruppo predefinito o un gruppo specificato in un `clusterrolebinding` o un `rolebinding`. Per ulteriori informazioni consulta Ruoli predefiniti e associazioni di ruoli nella documentazione di Kubernetes.

Ad esempio, il blocco YAML seguente contiene:

- Una sezione `mapRoles` che mappa l'istanza del nodo IAM ai gruppi Kubernetes in modo che i nodi possano registrarsi autonomamente con il cluster e il ruolo IAM `my-console-viewer-role` mappato a un gruppo Kubernetes in grado di visualizzare tutte le risorse Kubernetes per tutti i cluster. Per un elenco delle autorizzazioni del gruppo IAM e Kubernetes necessarie per il ruolo IAM `my-console-viewer-role`, consulta Autorizzazioni richieste.
- Una `mapUsers` sezione che mappa l'utente `admin` IAM dall' AWS account predefinito al `system:masters` Kubernetes gruppo e l'`my-user`utente di un AWS account diverso mappato a un Kubernetes gruppo in grado di visualizzare Kubernetes le risorse per uno spazio dei nomi specifico. Per un elenco delle autorizzazioni del gruppo IAM e Kubernetes necessarie per l'utente IAM `my-user`, consulta Autorizzazioni richieste.

Aggiungi o rimuovi righe in base alle esigenze e sostituisci tutti i valori *`example values`* con quelli desiderati.

```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this
 file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  mapRoles: |
    - groups:
      - system:bootstrappers
      - system:nodes
      rolearn: arn:aws:iam::111122223333:role/my-role
      username: system:node:{{EC2PrivateDNSName}}
```

```
    - groups:
      - eks-console-dashboard-full-access-group
      rolearn: arn:aws:iam::111122223333:role/my-console-viewer-role
      username: my-console-viewer-role
  mapUsers: |
    - groups:
      - system:masters
      userarn: arn:aws:iam::111122223333:user/admin
      username: admin
    - groups:
      - eks-console-dashboard-restricted-access-group
      userarn: arn:aws:iam::444455556666:user/my-user
      username: my-user
```

3. Salva il file ed esci dall'editor di testo.

Applica la **aws-authConfigMap** al cluster

La `aws-auth` `ConfigMap` viene creata e applicata automaticamente al cluster quando si crea un gruppo di nodi gestito o quando si crea un gruppo di nodi utilizzando `eksctl`. È inizialmente creata per consentire ai nodi di unirsi al cluster, ma questa `ConfigMap` può essere usata anche per aggiungere l'accesso RBAC (controllo degli accessi basato sul ruolo) a principali IAM. Se non sono stati avviati nodi autogestiti e non è stata applicata la `aws-auth` `ConfigMap` al cluster, è possibile farlo utilizzando la procedura seguente.

Applicazione di **aws-authConfigMap** al cluster

1. Controlla per vedere se hai già applicato la `aws-auth` `ConfigMap`.

   ```
   kubectl describe configmap -n kube-system aws-auth
   ```

   Se si verifica un errore di tipo "`Error from server (NotFound): configmaps "aws-auth" not found`", procedi con le fasi seguenti per applicare lo stock `ConfigMap`.

2. Scarica, modifica e applica la mappa di configurazione dell' AWS autenticatore.

   a. Scarica la mappa di configurazione.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/aws-auth-cm.yaml
   ```

b. Nel file `aws-auth-cm.yaml`, imposta il `rolearn` sul nome della risorsa Amazon (ARN) del ruolo IAM associato ai nodi. Per eseguire questa operazione, utilizza un editor di testo o sostituisci *`my-node-instance-role`* eseguendo il comando seguente:

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-role|' aws-auth-cm.yaml
```

Non modificare altre righe in questo file.

> **Important**
>
> L'ARN del ruolo non può includere un percorso, ad esempio `role/my-team/developers/my-role`. Il formato dell'ARN deve essere `arn:aws:iam::111122223333:role/my-role`. In questo esempio, `my-team/developers/` deve essere rimosso.

Puoi ispezionare gli output dello AWS CloudFormation stack per i tuoi gruppi di nodi e cercare i seguenti valori:

- InstanceRoleARN — Per i gruppi di nodi creati con `eksctl`
- NodeInstanceRuolo: per i gruppi di nodi creati con i AWS CloudFormation modelli forniti da Amazon EKS nel AWS Management Console

c. Applica la configurazione. L'esecuzione di questo comando potrebbe richiedere alcuni minuti.

```
kubectl apply -f aws-auth-cm.yaml
```

> **Note**
>
> Se ricevi qualsiasi altro errore di tipo di risorsa o autorizzazione, consulta la sezione Accesso negato o non autorizzato (`kubectl`) nell'argomento relativo alla risoluzione dei problemi.

3. Guarda lo stato dei nodi e attendi che raggiungano lo stato `Ready`.

```
kubectl get nodes --watch
```

Inserisci `Ctrl+C` per tornare a un prompt della shell.

Autentica gli utenti del tuo cluster da un provider di OpenID Connect identità

Amazon EKS supporta l'utilizzo di OpenID Connect (OIDC) provider di identità come metodo per autenticare gli utenti nel tuo cluster. OIDCi provider di identità possono essere utilizzati con o in alternativa a AWS Identity and Access Management (IAM). Per ulteriori informazioni sull'utilizzo di IAM, consultare the section called “Concedi l'accesso alle API Kubernetes”. Dopo aver configurato l'autenticazione per il cluster, potrai creare `roles` e `clusterroles` Kubernetes per assegnare autorizzazioni ai ruoli e quindi associare i ruoli alle identità utilizzando `rolebindings` e `clusterrolebindings` Kubernetes. Per ulteriori informazioni, consulta Utilizzo dell'autorizzazione RBAC nella documentazione di Kubernetes.

Considerazioni

- È possibile associare un provider di identità OIDC al cluster.
- Kubernetes non fornisce un provider di identità OIDC. È possibile utilizzare un provider di identità OIDC pubblico esistente oppure eseguire il proprio provider di identità. Per un elenco dei provider certificati, consultare OpenID Certification sul sito OpenID.
- L'URL dell'emittente del provider di identità OIDC deve essere accessibile pubblicamente, in modo che Amazon EKS possa individuare le chiavi di firma. Amazon EKS non supporta i provider di identità OIDC con certificati autofirmati.
- Non è possibile disabilitare l'autenticazione IAM nel cluster, perché sarà ancora necessaria per aggiungere i nodi a un cluster.
- Un cluster Amazon EKS deve comunque essere creato da un principale AWS IAM, anziché da un utente di un provider di OIDC identità. Questo perché il creatore del cluster interagisce con le API Amazon EKS, piuttosto che con le API Kubernetes.
- OIDCgli utenti autenticati dal provider di identità vengono elencati nel registro di controllo del cluster se CloudWatch i log sono attivati per il piano di controllo. Per ulteriori informazioni, consulta Abilitazione e disabilitazione dei log del piano di controllo.
- Non è possibile accedere a AWS Management Console con un account di un provider. OIDC Puoi visualizzare Kubernetes le risorse nella console solo accedendo AWS Management Console con un AWS Identity and Access Management account.

Associazione di un provider di identità OIDC

Prima di associare un provider di identità OIDC al cluster, è necessario ottenere dal provider le seguenti informazioni:

URL dell'emittente

L'URL del provider di identità OIDC che consente al server API di individuare le chiavi di firma pubbliche per la verifica dei token. L'URL deve iniziare con `https://` e deve corrispondere alla richiesta `iss` nei token ID OIDC del provider. In conformità con lo standard OIDC, i componenti del percorso sono consentiti, ma i parametri di query non lo sono. In genere l'URL è costituito solo da un nome host, come `https://server.example.org` o `https://example.com`. Questo URL dovrebbe puntare al livello sottostante `.well-known/openid-configuration` e dovrà essere accessibile pubblicamente tramite Internet.

ID client (noto anche come pubblico)

L'ID per l'applicazione client che effettua richieste di autenticazione al provider di identità OIDC.

È possibile associare un provider di identità utilizzando `eksctl` o il AWS Management Console.

eksctl

Per associare un provider di identità OIDC al cluster utilizzando **`eksctl`**

1. Crea un file denominato *`associate-identity-provider.yaml`* con i seguenti contenuti. Sostituisci i *`example values`* con i valori in tuo possesso. I valori nella sezione `identityProviders` vengono ottenuti dal provider di identità OIDC. I valori sono obbligatori solo per il `name`, `type`, `issuerUrl`, e `clientId` impostazioni in `identityProviders`.

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: your-region-code

identityProviders:
  - name: my-provider
```

```
    type: oidc
    issuerUrl: https://example.com
    clientId: kubernetes
    usernameClaim: email
    usernamePrefix: my-username-prefix
    groupsClaim: my-claim
    groupsPrefix: my-groups-prefix
    requiredClaims:
      string: string
    tags:
      env: dev
```

> **Important**
>
> Non specificare `system:`, o qualsiasi parte di quella stringa, per `groupsPrefix` o `usernamePrefix`.

2. Creare il provider.

```
eksctl associate identityprovider -f associate-identity-provider.yaml
```

3. Per utilizzare `kubectl` con il tuo cluster e il provider di identità OIDC, consulta la sezione Using `kubectl` nella documentazione di Kubernetes.

AWS Management Console

Per associare un provider di OIDC identità al cluster utilizzando il AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Seleziona il tuo cluster, quindi seleziona la scheda Accesso.
3. Nella sezione Provider di OIDC identità, seleziona Associate Identity Provider.
4. Nella pagina Associa provider di identità OIDC, inserire o selezionare le seguenti opzioni e quindi selezionare Associa.
   - In Nome, inserire un nome univoco per il provider.
   - Per URL dell'emittente, inserire l'URL per il provider. Questo URL deve essere accessibile in Internet.

- Per ID client, inserire l'ID client del provider di identità OIDC (noto anche come audience).
- Per Richiesta Nome Utente, inserire la richiesta da utilizzare come nome utente.
- Per Richiesta di Gruppi, inserire la richiesta da utilizzare come gruppo dell'utente.
- (Opzionale) Selezionare Opzioni avanzate, inserire o selezionare le seguenti informazioni.
  - Prefisso Nome utente: inserire un prefisso da anteporre alle richieste del nome utente. Il prefisso viene anteposto alle richieste di nome utente per evitare conflitti con i nomi esistenti. Se non si fornisce un valore e il nome utente è un valore diverso da `email`, il prefisso viene impostato per impostazione predefinita sul valore per URL dell'emittente. È possibile utilizzare il valore `-` per disabilitare tutti i prefissi. Non specificare `system:` o qualsiasi parte di quella stringa.
  - Prefisso Gruppi: inserire un prefisso da anteporre alle richieste di gruppo. Il prefisso viene anteposto alle richieste di gruppo per evitare conflitti con i nomi esistenti (ad esempio `system: groups`). Ad esempio, il valore `oidc:` crea nomi di gruppo come `oidc:engineering` e `oidc:infra`. Non specificare `system:` o qualsiasi parte di quella stringa..
  - Richieste necessarie: selezionare Aggiungi richiesta e inserire una o più coppie di valori chiave che descrivono le richieste necessarie nel token ID client. Le coppie descrivono le richieste necessarie nel Token ID. Se impostato, ogni richiesta viene verificata per essere presente nel token ID con un valore corrispondente.

5. Per utilizzare `kubectl` con il tuo cluster e il provider di identità OIDC, consulta la sezione Using `kubectl` nella documentazione di Kubernetes.

Dissociazione di un provider di identità OIDC dal cluster

Se si dissocia un provider di identità OIDC dal cluster, gli utenti inclusi nel provider non potranno più accedere al cluster. Tuttavia, è comunque possibile accedere al cluster con principali IAM.

Per dissociare un provider di identità OIDC dal cluster utilizzando la AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nella sezione Provider di identità OIDC, seleziona Dissocia, inserisci il nome del provider di identità e quindi seleziona `Disassociate`.

Policy IAM di esempio

Se si desidera impedire che un provider di identità OIDC venga associato a un cluster, creare e associare le seguenti policy IAM agli account IAM degli amministratori Amazon EKS. Per ulteriori informazioni, consultare Creazione di policy IAM e Aggiunta autorizzazioni di identità IAM nella Guida per l'utente di IAM e Operazioni, risorse e chiavi di condizione per Amazon Elastic Kubernetes Service in Service Authorization Reference.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "denyOIDC",
            "Effect": "Deny",
            "Action": [
                "eks:AssociateIdentityProviderConfig"
            ],
            "Resource": "arn:aws:eks:us-west-2.amazonaws.com:111122223333:cluster/*"

        },
        {
            "Sid": "eksAdmin",
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        }
    ]
}
```

La seguente policy di esempio consente l'associazione del provider di identità OIDC se il `clientID` è `kubernetes` e la `issuerUrl` è `https://cognito-idp.us-west-2amazonaws.com/*`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCognitoOnly",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
```

```
            "Condition": {
                "StringNotLikeIfExists": {
                    "eks:issuerUrl": "https://cognito-idp.us-west-2.amazonaws.com/*"
                }
            }
        },
        {
            "Sid": "DenyOtherClients",
            "Effect": "Deny",
            "Action": "eks:AssociateIdentityProviderConfig",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-instance",
            "Condition": {
                "StringNotEquals": {
                    "eks:clientId": "kubernetes"
                }
            }
        },
        {
            "Sid": "AllowOthers",
            "Effect": "Allow",
            "Action": "eks:*",
            "Resource": "*"
        }
    ]
}
```

Provider di identità OIDC convalidati dai partner

Amazon EKS dispone di una rete di partner che offrono supporto per i provider di identità OIDC compatibili. Per informazioni dettagliate su come integrare il provider di identità con Amazon EKS, consulta la documentazione dei partner riportata di seguito.

Partner	Product	Documentazione
PingIdentity	PingOne per Enterprise	Istruzioni di installazione

Amazon EKS mira a dare all'utente una vasta gamma di opzioni per coprire tutti i casi d'uso. Se sviluppi un provider di identità compatibile con OIDC supportato commercialmente che non è elencato qui, ti invitiamo a contattare il nostro team dei partner all'indirizzo aws-container-partners@amazon.com per ulteriori informazioni.

Creazione o aggiornamento di un file **kubeconfig** per un cluster Amazon EKS

In questo argomento viene creato un file `kubeconfig` per il cluster (o ne viene aggiornato uno esistente).

Lo strumento della linea di comando `kubectl` utilizza le informazioni di configurazione nei file `kubeconfig` per comunicare con il server API di un cluster. Per ulteriori informazioni, consulta la sezione Organizing Cluster Access Using kubeconfig Files (Organizzazione dell'accesso ai cluster utilizzando i file kubeconfig) nella documentazione di Kubernetes.

Amazon EKS utilizza il comando `aws eks get-token` con `kubectl` per l'autenticazione del cluster. Per impostazione predefinita, AWS CLI utilizza le stesse credenziali restituite con il seguente comando:

```
aws sts get-caller-identity
```

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.

- Un ruolo o un utente IAM con l'autorizzazione a utilizzare l'azione API `eks:DescribeCluster` per il cluster che specifichi. Per ulteriori informazioni, consulta Esempi di policy basate su identità Amazon EKS. Se utilizzi un'identità del tuo provider OpenID Connect per accedere al cluster, consulta la sezione Using `kubectl` nella documentazione di Kubernetes per creare o aggiornare il file `kube config`.

Creazione automatica del file **`kubeconfig`**

Prerequisiti

- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.
- Autorizzazione a utilizzare l'operazione API `eks:DescribeCluster` per il cluster specificato. Per ulteriori informazioni, consulta Esempi di policy basate su identità Amazon EKS.

Per creare il **`kubeconfig`** file con AWS CLI

1. Crea o aggiorna un file `kubeconfig` per il cluster. *`Sostituisci il codice regionale con il nome del tuo cluster.`*

   ```
   aws eks update-kubeconfig --region region-code --name my-cluster
   ```

   Per impostazione predefinita, il file di configurazione risultante viene creato nel percorso `kubeconfig` predefinito (`.kube`) nella home directory o unito a un file `config` esistente in quel percorso. È possibile specificare un altro percorso con l'opzione **`--kubeconfig`**.

   Puoi specificare un ARN del ruolo IAM con l'opzione **`--role-arn`** da utilizzare per l'autenticazione quando esegui comandi `kubectl`. Altrimenti, viene utilizzato il principale

IAM nella catena di credenziali predefinita o SDK. AWS CLI Puoi visualizzare la tua identità predefinita AWS CLI o SDK eseguendo il comando. `aws sts get-caller-identity`

Per tutte le opzioni disponibili, esegui il comando `aws eks update-kubeconfig help` o consulta la sezione `update-kubeconfig` nella Documentazione di riferimento ai comandi della AWS CLI .

2. Prova la configurazione.

```
kubectl get svc
```

Di seguito viene riportato un output di esempio:

```
NAME             TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
svc/kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   1m
```

Se ricevi qualsiasi altro errore di tipo di risorsa o autorizzazione, consulta la sezione Accesso negato o non autorizzato (`kubectl`) nell'argomento relativo alla risoluzione dei problemi.

Concedi ai carichi di lavoro Kubernetes l'accesso all'utilizzo degli account di servizio AWSKubernetes

Un account del servizio Kubernetes fornisce un'identità per i processi eseguiti in un Pod. Per ulteriori informazioni, consulta Gestione degli account di servizio nella documentazione di Kubernetes. Se hai Pod bisogno di accedere ai AWS servizi, puoi mappare l'account del servizio a un' AWS Identity and Access Management identità per concedere tale accesso. Per ulteriori informazioni, consulta Ruoli IAM per gli account di servizio.

Token dell'account di servizio

Per impostazione predefinita, la funzionalità `BoundServiceAccountTokenVolume` è abilitata nelle versioni Kubernetes. Questa funzionalità migliora la sicurezza dei token dell'account di servizio, consentendo ai carichi di lavoro in esecuzione su Kubernetes di richiedere token Web JSON associati a pubblico, tempo e chiavi. I token dell'account di servizio scadono entro un'ora, Nelle versioni di Kubernetes precedenti, i token non avevano una data di scadenza. Di conseguenza, i client che si basano su questi token devono aggiornarli entro un'ora. Gli SDK del client Kubernetes seguenti aggiornano automaticamente i token entro il periodo di tempo richiesto:

- Go versione `0.15.7` e successive
- Python versione `12.0.0` e successive
- Java versione `9.0.0` e successive
- JavaScript versione `0.10.3` e successive
- Ramo `master` di Ruby
- Haskell versione `0.3.0.0`
- C# versione `7.0.5` e successive

Se il carico di lavoro utilizza una versione precedente del client, è necessario aggiornarla. Per consentire una migrazione agevole dei client verso i più recenti token dell'account di servizio con limite di tempo, Kubernetes proroga il periodo di scadenza dei token, rispetto a quello predefinito di un'ora. Per i cluster Amazon EKS, il periodo di scadenza è prorogato a 90 giorni. Il server API Kubernetes del cluster Amazon EKS rifiuta le richieste con token più vecchi di 90 giorni. Consigliamo di controllare le applicazioni e le relative dipendenze per assicurarti che gli SDK del client Kubernetes corrispondano o siano successivi alle versioni elencate sopra.

Quando il server API riceve richieste con token più vecchie di un'ora, le annota nel log eventi di controllo API con `annotations.authentication.k8s.io/stale-token`. Il valore dell'annotazione è simile a quello riportato di seguito:

```
subject: system:serviceaccount:common:fluent-bit, seconds after warning threshold:
 4185802.
```

Se nel cluster è abilitata la registrazione del piano di controllo (control-plane), le annotazioni di trovano nei registri di controllo. Puoi utilizzare la seguente query CloudWatch Logs Insights per identificare tutti gli utenti del Pods tuo cluster Amazon EKS che utilizzano token obsoleti:

```
fields @timestamp
| filter @logStream like /kube-apiserver-audit/
| filter @message like /seconds after warning threshold/
| parse @message "subject: *, seconds after warning threshold:*\"" as subject,
 elapsedtime
```

L'`subject` si riferisce all'account di servizio utilizzato dal Pod. `elapsedtime` indica in secondi il tempo trascorso dalla lettura dell'ultimo token. Le richieste al server API vengono negate quando `elapsedtime` supera i 90 giorni (7.776.000 secondi). Per utilizzare una delle versioni elencate sopra

e aggiornare automaticamente il token, devi aggiornare in modo proattivo l'SDK client Kubernetes delle applicazioni. Se il token dell'account di servizio utilizzato è prossimo a 90 giorni e non hai tempo sufficiente per aggiornare le versioni dell'SDK client prima della scadenza del token, puoi terminare i Pods esistenti e creare nuovi pod. Ciò comporta una ridefinizione del token dell'account di servizio, concedendo agli utenti altri 90 giorni per aggiornare gli SDK della versione client.

Se il Pod fa parte di un'implementazione, è consigliabile terminare i Pods eseguendo un rollout in modo da mantenere un'elevata disponibilità. Per eseguire questa operazione, utilizza il comando seguente. Sostituisci *my-deployment* con il nome della tua implementazione.

```
kubectl rollout restart deployment/my-deployment
```

Componenti aggiuntivi del cluster

I componenti aggiuntivi del cluster seguenti sono stati aggiornati per utilizzare gli SDK client Kubernetes che riconfigurano automaticamente i token dell'account di servizio. Assicurati che le versioni elencate, o versioni successive, siano installate sul tuo cluster.

- Amazon VPC CNI plugin for Kubernetes e plugin dell'helper di parametri versione `1.8.0` e successive. Per verificare la versione corrente o aggiornarla, consulta e. Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS cni-metrics-helper
- CoreDNS versione `1.8.4` e successive. Per verificare la versione attuale o aggiornarla, consulta Utilizzo del componente aggiuntivo CoreDNS di Amazon EKS.
- AWS Load Balancer Controller versione `2.0.0` e successive. Per verificare la versione attuale o aggiornarla, consulta Che cosa è la AWS Load Balancer Controller?.
- Versione `kube-proxy` attuale. Per verificare la versione attuale o aggiornarla, consulta Utilizzo del componente aggiuntivo Kubernetes `kube-proxy`.
- AWS per la versione Fluent Bit `2.25.0` o successiva. Per aggiornare la versione corrente, consulta Versioni su GitHub.
- Versione dell'immagine di Fluentd 1.14.6-1.2 o versione successiva e plugin di filtro Fluentd per la versione dei metadati Kubernetes 2.11.1 o successive.

Concessione AWS Identity and Access Management delle autorizzazioni ai carichi di lavoro sui cluster Amazon Elastic Kubernetes Service

Amazon EKS offre due modi per concedere AWS Identity and Access Management le autorizzazioni ai carichi di lavoro eseguiti nei cluster Amazon EKS: ruoli IAM per gli account di servizio e EKS Pod Identities.

Ruoli IAM per gli account di servizio

IAM roles for service accounts (IRSA) configura le applicazioni Kubernetes in esecuzione AWS con autorizzazioni IAM granulari per accedere a varie altre risorse come i bucket Amazon AWS S3, le tabelle Amazon DynamoDB e altro ancora. Puoi eseguire più applicazioni contemporaneamente nello stesso cluster Amazon EKS e assicurarti che ogni applicazione disponga solo del set minimo di autorizzazioni di cui ha bisogno. IRSA è stato creato per supportare varie opzioni di Kubernetes distribuzione supportate da AWS Amazon EKS, Amazon EKS Anywhere e Kubernetes cluster autogestiti su istanze Amazon EC2. Servizio Red Hat OpenShift su AWS Pertanto, IRSA è stato creato utilizzando un AWS servizio fondamentale come IAM e non ha assunto alcuna dipendenza diretta dal servizio Amazon EKS e dall'API EKS. Per ulteriori informazioni, consulta Ruoli IAM per gli account di servizio.

EKS Pod Identity

EKS Pod Identity offre agli amministratori dei cluster un flusso di lavoro semplificato per l'autenticazione delle applicazioni per accedere a varie altre AWS risorse come bucket Amazon S3, tabelle Amazon DynamoDB e altro ancora. EKS Pod Identity è solo per EKS e, di conseguenza, semplifica il modo in cui gli amministratori dei cluster possono configurare le applicazioni Kubernetes per ottenere le autorizzazioni IAM. Queste autorizzazioni possono ora essere facilmente configurate con meno passaggi direttamente tramite l' AWS Management Console API EKS e AWS CLI non è necessario eseguire alcuna azione all'interno del cluster su nessun oggetto. Kubernetes Gli amministratori del cluster non devono passare dai servizi EKS ai servizi IAM o utilizzare operazioni IAM privilegiate per configurare le autorizzazioni richieste dalle applicazioni. I ruoli IAM possono ora essere utilizzati su più cluster senza la necessità di aggiornare la policy di attendibilità dei ruoli durante la creazione di nuovi cluster. Le credenziali IAM fornite da EKS Pod Identity includono tag di sessione dei ruoli, con attributi come nome del cluster, namespace, nome dell'account di servizio. I tag delle sessioni di ruolo consentono agli amministratori di creare un singolo ruolo in grado di funzionare su più account di servizio, consentendo l'accesso alle AWS risorse in base ai tag corrispondenti. Per ulteriori informazioni, consulta EKS Pod Identity.

Confronto tra EKS Pod Identity e IRSA

A un livello superiore, sia EKS Pod Identity che IRSA concedono autorizzazioni IAM alle applicazioni in esecuzione su cluster Kubernetes. Ma sono fondamentalmente diversi nel modo in cui vengono configurati, nei limiti supportati e nelle funzionalità abilitate. Di seguito, confrontiamo alcuni degli aspetti chiave di entrambe le soluzioni.

	EKS Pod Identity	IRSA
Estensibilità dei ruoli	È necessario configurare ogni ruolo una volta per instaurare un rapporto di attendibilità con il principale di servizio Amazon EKS introdotto di recente, `pods.eks.amazonaws.com` . Dopo questo passaggio una tantum, non è necessario aggiornare la policy di attendibilità del ruolo ogni volta che viene utilizzato in un nuovo cluster.	È necessario aggiornare la policy di attendibilità del ruolo IAM con il nuovo endpoint del provider OIDC del cluster EKS ogni volta che si desidera utilizzare il ruolo in un nuovo cluster.
Scalabilità del cluster	EKS Pod Identity non richiede agli utenti di configurare il provider IAM OIDC, quindi questo limite non si applica.	Ogni cluster EKS ha un URL emittente OpenID Connect (OIDC) associato. Per utilizzare IRSA, è necessario creare un provider OpenID Connect univoco per ogni cluster EKS in IAM. IAM ha un limite globale predefinito di 100 provider OIDC per ciascun Account AWS. Se prevedi di avere più di 100 cluster EKS per ciascuno Account AWS con IRSA, raggiungerai il limite del provider IAMOIDC.

	EKS Pod Identity	IRSA
Scalabilità dei ruoli	EKS Pod Identity non richiede agli utenti di definire una relazione di attendibilità tra il ruolo IAM e l'account di servizio nella policy di attendibilità, quindi questo limite non viene applicato.	In IRSA, definisci la relazione di attendibilità tra un ruolo IAM e un account di servizio nella policy di attendibilità del ruolo. Per impostazione predefinita, la lunghezza della dimensione della policy di attendibilità è `2048`. Ciò significa che in genere è possibile definire 4 relazioni di attendibilità in un'unica policy di attendibilità. Sebbene sia possibile aumentare il limite di lunghezza della policy di attendibilità, in genere si è limitati a un massimo di 8 relazioni di attendibilità all'interno di una singola policy di attendibilità.

	EKS Pod Identity	IRSA
Riutilizzabilità dei ruoli	AWS STS le credenziali temporanee fornite da EKS Pod Identity includono tag di sessione del ruolo, come il nome del cluster, lo spazio dei nomi, il nome dell'account del servizio. I tag di sessione dei ruoli permettono agli amministratori di creare un singolo ruolo IAM che può essere utilizzato con più account di servizio, con diverse autorizzazioni effettive, consentendo l'accesso alle risorse AWS in base ai tag ad essi collegati . Questo ruolo è chiamato anche controllo degli accessi basato su attributi (ABAC). Per ulteriori informazioni, consulta Definire le autorizzazioni per fare in modo che le associazioni EKS Pod Identity assumano i ruoli in base ai tag.	AWS STS i tag di sessione non sono supportati. È possibile riutilizzare un ruolo tra i cluster, ma ogni pod riceve tutte le autorizzazioni del ruolo.
Ambienti supportati	EKS Pod Identity è disponibile solo su Amazon EKS.	È possibile utilizzare IRSA come Amazon EKS, Amazon EKS Anywhere e Kubernetes cluster autogestiti su istanze Amazon EC2. Servizio Red Hat OpenShift su AWS

	EKS Pod Identity	IRSA
Versioni EKS supportate	EKS Kubernetes versioni `1.24` o successive. Per le versioni delle piattaforme specifiche, consulta Versioni del cluster EKS Pod Identity.	Tutte le versioni del cluster EKS supportate.

EKS Pod Identity

Le applicazioni nei contenitori Pod a's possono utilizzare un AWS SDK o AWS CLI effettuare richieste API per Servizi AWS utilizzare le autorizzazioni AWS Identity and Access Management (IAM). Le applicazioni devono firmare le proprie richieste AWS API con AWS credenziali.

Le associazioni EKS Pod Identity forniscono la possibilità di gestire le credenziali per le applicazioni, analogamente al modo in cui i profili di istanza Amazon EC2 forniscono le credenziali alle istanze Amazon EC2. Invece di creare e distribuire AWS le tue credenziali nei contenitori o utilizzare il ruolo dell'istanza Amazon EC2, associ un ruolo IAM a un account di servizio e configuri Kubernetes Pods il tuo per l'utilizzo dell'account di servizio.

Ogni associazione EKS Pod Identity associa un ruolo a un account di servizio in uno spazio dei nomi nel cluster specificato. Se hai la stessa applicazione in più cluster, puoi creare associazioni identiche in ogni cluster senza modificare la policy di attendibilità del ruolo.

Se un pod utilizza un account di servizio con un'associazione, Amazon EKS imposta le variabili di ambiente nei container del pod. Le variabili di ambiente configurano gli AWS SDK, tra cui, per utilizzare le AWS CLI credenziali EKS Pod Identity.

Vantaggi delle associazioni EKS Pod Identity

Le associazioni EKS Pod Identity offrono i seguenti vantaggi:

- Privilegio minimo: è possibile definire l'ambito delle autorizzazioni IAM per un account di servizio; solo i Pods che utilizzano tale account avranno accesso alle autorizzazioni definite. Questa caratteristica elimina anche la necessità di soluzioni di terze parti, ad esempio `kiam` o `kube2iam`.
- Isolamento delle credenziali: un container del Pod's può recuperare solo le credenziali per il ruolo IAM associato all'account del servizio che utilizza. Un container non ha mai accesso alle

credenziali utilizzate da altri container in Pods. Quando si utilizzano le associazioni Pod Identity, i container Pod's hanno anche le autorizzazioni assegnate al ruolo IAM del nodo Amazon EKS, a meno che non blocchi l'accesso del Pod al servizio di metadati di istanza (IMDS) di Amazon EC2. Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.

- Verificabilità: la registrazione degli accessi e degli eventi è disponibile AWS CloudTrail per facilitare il controllo retrospettivo.

EKS Pod Identity è un metodo più semplice di Ruoli IAM per gli account di servizio, in quanto questo metodo non utilizza provider di identità OIDC. EKS Pod Identity presenta i seguenti miglioramenti:

- Operazioni indipendenti: in molte organizzazioni, la creazione di provider di identità OIDC è una responsabilità di team diversi rispetto all'amministrazione dei cluster Kubernetes. EKS Pod Identity offre una netta separazione dei compiti, in cui tutta la configurazione delle associazioni EKS Pod Identity viene eseguita in Amazon EKS e tutta la configurazione delle autorizzazioni IAM viene eseguita in IAM.
- Riusabilità: EKS Pod Identity utilizza un singolo principale IAM anziché principali separati per ogni cluster utilizzato dai ruoli IAM per gli account di servizio. L'amministratore IAM aggiunge il seguente principale alla policy di attendibilità di qualsiasi ruolo per renderlo utilizzabile da parte delle associazioni EKS Pod Identity.

```
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            }
```

- Scalabilità: ogni set di credenziali temporanee viene assunto dal EKS Auth servizio in EKS Pod Identity, anziché da ogni AWS SDK eseguito in ciascun pod. Quindi, si esegue Amazon EKS Pod Identity Agent su ogni nodo rilasciando le credenziali agli SDK. Pertanto, il carico si riduce a una volta per ogni nodo e non viene duplicato in ogni pod. Per ulteriori dettagli del processo, consulta Come funziona EKS Pod Identity.

Per ulteriori informazioni su come confrontare le due alternative, consulta Concedi ai carichi di lavoro Kubernetes l'accesso all'utilizzo degli account di servizio AWSKubernetes.

Panoramica sulla configurazione delle associazioni EKS Pod Identity

Attiva le associazioni EKS Pod Identity completando le seguenti procedure:

1. Configurazione di Amazon EKS Pod Identity Agent: questa procedura viene completata una sola volta per ogni cluster.
2. Configura un account Kubernetes di servizio per assumere un ruolo IAM con EKS Pod Identity: completa questa procedura per ogni set univoco di autorizzazioni che desideri abbia un'applicazione.

3. PodsConfigurare l'utilizzo di un account Kubernetes di servizio— Completate questa procedura per ognuno a Pod cui è necessario accedere. Servizi AWS
4. Usa un AWS SDK supportato— Conferma che il carico di lavoro utilizzi un AWS SDK di una versione supportata e che utilizzi la catena di credenziali predefinita.

Considerazioni su EKS Pod Identity

- È possibile associare un ruolo IAM a ciascun account di servizio Kubernetes in ogni cluster. È possibile modificare il ruolo mappato all'account di servizio modificando l'associazione EKS Pod Identity.
- È possibile associare solo ruoli che fanno parte dello stesso Account AWS cluster. È possibile delegare l'accesso da un altro account al ruolo di questo account configurato per l'utilizzo delle associazioni EKS Pod Identity. Per un tutorial sulla delega dell'accesso e`AssumeRole`, consulta Delegare l'accesso tra AWS account utilizzando i ruoli IAM nella IAM User Guide.
- EKS Pod Identity Agent è obbligatorio. Funziona come un `DaemonSet` Kubernetes sui nodi e fornisce le credenziali solo ai pod sul nodo su cui viene eseguito. Per ulteriori informazioni sulla compatibilità con EKS Pod Identity Agent, consulta la sezione seguente Restrizioni di EKS Pod Identity.
- EKS Pod Identity Agent utilizza la `hostNetwork` del nodo e utilizza la porta `80` e la porta `2703` su un indirizzo locale del collegamento sul nodo. Questo indirizzo è `169.254.170.23` per IPv4 e `[fd00:ec2::23]` per i cluster IPv6.

  Se `IPv6` disabiliti gli indirizzi o impedisci in altro modo gli indirizzi `IPv6` IP di localhost, l'agente non può avviarsi. Per avviare l'agente su nodi che non possono essere utilizzati`IPv6`, segui i passaggi indicati Disabilita `IPv6` nell'EKS Pod Identity Agent per disabilitare la `IPv6` configurazione.

Versioni del cluster EKS Pod Identity

Per utilizzare le associazioni EKS Pod Identity, il cluster deve avere una versione della piattaforma uguale o successiva a quella elencata nella tabella seguente o una versione Kubernetes successiva a quelle elencate nella tabella.

Versione Kubernetes	Versione della piattaforma
`1.30`	`eks.2`
`1.29`	`eks.1`
`1.28`	`eks.4`
`1.27`	`eks.8`
`1.26`	`eks.9`
`1.25`	`eks.10`
`1.24`	`eks.13`

Versioni aggiuntive compatibili con EKS Pod Identity

> **Important**
>
> Per utilizzare EKS Pod Identity con un componente aggiuntivo EKS, è necessario creare manualmente l'associazione EKS Pod Identity. Non scegliete un ruolo IAM nella configurazione del componente aggiuntivo in AWS Management Console, tale ruolo viene utilizzato solo con IRSA.

I componenti aggiuntivi Amazon EKS e i componenti aggiuntivi autogestiti che richiedono credenziali IAM possono utilizzare EKS Pod Identity, IRSA o il ruolo dell'istanza. L'elenco dei componenti aggiuntivi che utilizzano credenziali IAM che supportano EKS Pod Identity è:

- Amazon VPC CNI plugin for Kubernetes`1.15.5-eksbuild.1`o più tardi

- AWS Load Balancer Controller`2.7.0`o più tardi. Tieni presente che non AWS Load Balancer Controller è disponibile come componente aggiuntivo EKS, ma è disponibile come componente aggiuntivo autogestito.

Restrizioni di EKS Pod Identity

Le associazioni EKS Pod Identity sono disponibili in:

- Versioni del cluster Amazon EKS elencate nell'argomento precedente Versioni del cluster EKS Pod Identity.
- Nodi worker nel cluster che sono istanze Amazon EC2 di Linux.

Le associazioni EKS Pod Identity non sono disponibili nelle:

- Regioni cinesi.
- AWS GovCloud (US).
- AWS Outposts.
- Amazon EKS Anywhere.
- Cluster Kubernetes creati ed eseguiti su Amazon EC2. I componenti di EKS Pod Identity sono disponibili solo su Amazon EKS.

Non puoi utilizzare le associazioni EKS Pod Identity con:

- Pod che vengono eseguiti ovunque, tranne che su istanze Amazon EC2 di Linux. I pod Linux e Windows su AWS Fargate (Fargate) cui vengono eseguiti non sono supportati. I pod che vengono eseguiti su istanze Amazon EC2 di Windows non sono supportati.
- Componenti aggiuntivi di Amazon EKS che richiedono le credenziali IAM. I componenti aggiuntivi EKS possono invece utilizzare solo ruoli IAM per account di servizio. L'elenco dei componenti aggiuntivi EKS che utilizzano le credenziali IAM include:
  - I driver di CSI storage: EBS CSI, EFS CSI, driver CSI Amazon FSx for Lustre, driver CSI Amazon FSx per ONTAP, driver CSI Amazon NetApp FSx per OpenZFS, driver CSI Amazon File AWS Cache, Secrets and Configuration Provider (ASCP) per il driver CSI Secrets Store Kubernetes

> **Note**
>
> Se questi controller, driver e plug-in vengono installati come componenti aggiuntivi autogestiti anziché componenti aggiuntivi EKS, supportano le identità EKS Pod purché vengano aggiornati per utilizzare gli AWS SDK più recenti.

Come funziona EKS Pod Identity

Le associazioni Amazon EKS Pod Identity offrono la possibilità di gestire le credenziali per le applicazioni, in modo simile a come i profili di istanza di Amazon EC2 forniscono le credenziali alle istanze Amazon EC2.

Amazon EKS Pod Identity fornisce le credenziali per i tuoi carichi di lavoro con un'API EKS Auth aggiuntiva e un pod di agenti che viene eseguito su ogni nodo.

Nei tuoi componenti aggiuntivi, come i componenti aggiuntivi Amazon EKS e i controller autogestiti, gli operatori e altri componenti aggiuntivi, l'autore deve aggiornare il proprio software per utilizzare gli SDK più recenti. AWS Per l'elenco della compatibilità tra EKS Pod Identity e i componenti aggiuntivi prodotti da Amazon EKS, consulta la sezione precedente Restrizioni di EKS Pod Identity.

Utilizzo delle associazioni EKS Pod Identity nel codice

Nel tuo codice, puoi utilizzare gli SDK per accedere ai AWS servizi. AWS Scrivi codice per creare un client per un AWS servizio con un SDK e, per impostazione predefinita, l'SDK cerca le credenziali da utilizzare in una catena di AWS Identity and Access Management posizioni. Dopo aver trovato credenziali valide, la ricerca viene interrotta. Per ulteriori informazioni sulle posizioni predefinite utilizzate, consulta la catena di fornitori di credenziali nella Guida di riferimento agli AWS SDK and Tools.

Le associazioni EKS Pod Identity sono state aggiunte al provider di credenziali del container, che viene cercato in un passaggio nella catena di credenziali predefinita. Se i carichi di lavoro utilizzano attualmente credenziali che si trovano all'inizio della catena di credenziali, quest'ultime continuano a essere utilizzate anche se configuri un'associazione EKS Pod Identity per lo stesso carico di lavoro. In questo modo è possibile migrare in sicurezza da altri tipi di credenziali creando l'associazione prima di rimuovere le vecchie credenziali.

Il provider di credenziali del container assegna credenziali temporanee da un agente che viene eseguito su ogni nodo. In Amazon EKS, l'agente è Amazon EKS Pod Identity Agent e su Amazon

Elastic Container Service l'agente è `amazon-ecs-agent`. Gli SDK utilizzano variabili di ambiente per individuare l'agente a cui connettersi.

Al contrario, i ruoli IAM per gli account di servizio forniscono un token di identità Web che l' AWS SDK deve utilizzare per lo scambio. AWS Security Token Service `AssumeRoleWithWebIdentity`

Come funziona EKS Pod Identity Agent con un Pod

1. Quando Amazon EKS avvia un nuovo pod che utilizza un account di servizio con un'associazione EKS Pod Identity, il cluster aggiunge il seguente contenuto al manifesto Pod:

```
    env:
    - name: AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE
      value: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/eks-pod-
identity-token"
    - name: AWS_CONTAINER_CREDENTIALS_FULL_URI
      value: "http://169.254.170.23/v1/credentials"
    volumeMounts:
    - mountPath: "/var/run/secrets/pods.eks.amazonaws.com/serviceaccount/"
      name: eks-pod-identity-token
  volumes:
  - name: eks-pod-identity-token
    projected:
      defaultMode: 420
      sources:
      - serviceAccountToken:
          audience: pods.eks.amazonaws.com
          expirationSeconds: 86400 # 24 hours
          path: eks-pod-identity-token
```

2. Kubernetes seleziona su quale nodo eseguire il pod. Quindi, Amazon EKS Pod Identity Agent sul nodo utilizza l'AssumeRoleForPodIdentityazione per recuperare le credenziali temporanee dall'API EKS Auth.
3. EKS Pod Identity Agent rende disponibili queste credenziali per gli AWS SDK che esegui all'interno dei tuoi contenitori.
4. Utilizza l'SDK nell'applicazione senza specificare un provider di credenziali per utilizzare la catena di credenziali predefinita. In alternativa, specifica il provider di credenziali del container. Per ulteriori informazioni sulle posizioni predefinite utilizzate, consulta la catena di fornitori di credenziali nella Guida di riferimento agli AWS SDK e agli strumenti.

5. L'SDK utilizza le variabili di ambiente per connettersi all'EKS Pod Identity Agent e recuperare le credenziali.

> **Note**
>
> Se i carichi di lavoro attualmente utilizzano credenziali che si trovano all'inizio della catena di credenziali, tali credenziali continueranno a essere utilizzate anche se configuri un'associazione EKS Pod Identity per lo stesso carico di lavoro.

Configurazione di Amazon EKS Pod Identity Agent

Le associazioni Amazon EKS Pod Identity offrono la possibilità di gestire le credenziali per le applicazioni, in modo simile a come i profili di istanza di Amazon EC2 forniscono le credenziali alle istanze Amazon EC2.

Amazon EKS Pod Identity fornisce le credenziali per i tuoi carichi di lavoro con un'API EKS Auth aggiuntiva e un pod di agenti che viene eseguito su ogni nodo.

Considerazioni

- **`IPv6`**

  Per impostazione predefinita, EKS Pod Identity Agent ascolta su un `IPv6` indirizzo `IPv4` and i pod per richiedere le credenziali. L'agente utilizza l'indirizzo IP di loopback (localhost) `IPv4` e l'indirizzo IP localhost `169.254.170.23` per. `[fd00:ec2::23]` `IPv6`

  Se `IPv6` disabiliti gli indirizzi o impedisci in altro modo gli indirizzi `IPv6` IP del localhost, l'agente non può avviarsi. Per avviare l'agente su nodi che non possono essere utilizzati`IPv6`, segui i passaggi indicati Disabilita `IPv6` nell'EKS Pod Identity Agent per disabilitare la `IPv6` configurazione.

Creazione di Amazon EKS Pod Identity Agent

Prerequisiti dell'agente

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS. La versione del cluster e la versione della piattaforma devono essere uguali o successive alle versioni elencate in Versioni del cluster EKS Pod Identity.

- Il ruolo del nodo dispone delle autorizzazioni per consentire all'agente di eseguire l'azione `AssumeRoleForPodIdentity` nell'API EKS Auth. Puoi utilizzare AWS politica gestita: AmazonEks WorkerNodePolicy o aggiungere una policy personalizzata simile alla seguente:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks-auth:AssumeRoleForPodIdentity"
            ],
            "Resource": "*"
        }
    ]
}
```

  Questa azione può essere limitata dai tag per ridurre i ruoli che possono essere assunti dai pod che utilizzano l'agente.

- I nodi possono raggiungere e scaricare immagini da Amazon ECR. L'immagine del container per il componente aggiuntivo si trova nei registri elencati in Registri delle immagini del container Amazon.

  Tieni presente che puoi modificare la posizione dell'immagine e fornire `imagePullSecrets` componenti aggiuntivi EKS nelle impostazioni di configurazione opzionali in AWS Management Console, e `--configuration-values` in. AWS CLI

- I nodi possono raggiungere l'API Amazon EKS Auth. Per i cluster privati, è necessario l'ingresso dell'`eks-auth`endpoint. AWS PrivateLink

AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi scegli il nome del cluster per cui configurare il componente aggiuntivo di EKS Pod Identity Agent.
3. Seleziona la scheda Componenti aggiuntivi.
4. Scegli Ottieni altri componenti aggiuntivi.

5. Seleziona la casella nella parte superiore destra di quella del componente aggiuntivo relativo a EKS Pod Identity Agent e scegli Avanti.
6. Nella pagina Configura le impostazioni dei componenti aggiuntivi selezionati, seleziona una versione qualsiasi nell'elenco a discesa Versione.
7. (Facoltativo) Espandi Impostazioni di configurazione facoltative per inserire una configurazione aggiuntiva. Ad esempio, puoi fornire una posizione alternativa per l'immagine del container e `ImagePullSecrets`. Lo JSON Schema con le chiavi accettate sono mostrate in Schema di configurazione del componente aggiuntivo.

   Inserisci le chiavi e i valori di configurazione in Valori di configurazione.
8. Seleziona Successivo.
9. Verifica che i pod di EKS Pod Identity Agent siano in esecuzione sul cluster.

   ```
   kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
   ```

   Di seguito viene riportato un output di esempio:

   ```
   eks-pod-identity-agent-gmqp7                                          1/1
    Running   1 (24h ago)   24h
   eks-pod-identity-agent-prnsh                                          1/1
    Running   1 (24h ago)   24h
   ```

   Ora puoi utilizzare le associazioni EKS Pod Identity nel cluster. Per ulteriori informazioni, consulta Configura un account Kubernetes di servizio per assumere un ruolo IAM con EKS Pod Identity.

AWS CLI

1. Esegui il comando seguente AWS CLI . Sostituisci `my-cluster` con il nome del cluster.

   ```
   aws eks create-addon --cluster-name my-cluster --addon-name eks-pod-identity-
   agent --addon-version v1.0.0-eksbuild.1
   ```

> **Note**
>
> EKS Pod Identity Agent non utilizza `service-account-role-arn` per ruoli IAM per account di servizio. È necessario fornire a EKS Pod Identity Agent le autorizzazioni nel ruolo del nodo.

2. Verifica che i pod di EKS Pod Identity Agent siano in esecuzione sul cluster.

```
kubectl get pods -n kube-system | grep 'eks-pod-identity-agent'
```

Di seguito viene riportato un output di esempio:

```
eks-pod-identity-agent-gmqp7                                          1/1
 Running   1 (24h ago)   24h
eks-pod-identity-agent-prnsh                                          1/1
 Running   1 (24h ago)   24h
```

Ora puoi utilizzare le associazioni EKS Pod Identity nel cluster. Per ulteriori informazioni, consulta Configura un account Kubernetes di servizio per assumere un ruolo IAM con EKS Pod Identity.

Aggiornamento di Amazon EKS Pod Identity Agent

Aggiorna il componente aggiuntivo del tipo Amazon EKS. Se non hai aggiunto il tipo di componente aggiuntivo Amazon EKS al cluster, consulta Creazione di Amazon EKS Pod Identity Agent.

AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi scegli il nome del cluster per cui configurare il componente aggiuntivo di EKS Pod Identity Agent.
3. Seleziona la scheda Componenti aggiuntivi.
4. Se è disponibile una nuova versione del componente aggiuntivo, in EKS Pod Identity Agent viene visualizzato un pulsante Aggiorna versione. Seleziona Aggiorna versione.

5. Nella pagina Configura Amazon EKS Pod Identity Agent, seleziona la nuova versione nell'elenco a discesa Versione.
6. Seleziona Salva modifiche.

   Il completamento dell'aggiornamento potrebbe richiedere alcuni secondi. Conferma quindi che la versione del componente aggiuntivo sia stata aggiornata controllando lo stato.

AWS CLI

1. Scopri qual è la versione del componente aggiuntivo attualmente installata sul cluster. Sostituisci *my-cluster* con il nome del cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --query "addon.addonVersion" --output text
   ```

   Di seguito viene riportato un output di esempio:

   ```
   v1.0.0-eksbuild.1
   ```

   È necessario creare il componente aggiuntivo prima di poterlo aggiornare con questa procedura.
2. Aggiorna il componente aggiuntivo utilizzando la AWS CLI. Se desideri utilizzare la AWS Management Console o il `eksctl` per aggiornare il componente aggiuntivo, consulta Aggiornamento di un componente aggiuntivo. Copia il comando seguente sul tuo dispositivo. Apporta le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato.

   - Sostituisci *my-cluster* con il nome del cluster.
   - Sostituisci la versione *v1.0.0-eksbuild.1* con quella desiderata.
   - Sostituire *111122223333* con l'ID account.
   - Esegui il comando seguente:

     ```
     aws eks update-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent --addon-version v1.0.0-eksbuild.1'
     ```

     Il completamento dell'aggiornamento potrebbe richiedere alcuni secondi.

3. Conferma che la versione del componente aggiuntivo sia stata aggiornata. Sostituisci *my-cluster* con il nome del cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name eks-pod-identity-agent
```

Il completamento dell'aggiornamento potrebbe richiedere alcuni secondi.

Di seguito viene riportato un output di esempio:

```
{
    "addon": {
        "addonName": "eks-pod-identity-agent",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.0.0-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/eks-pod-identity-agent/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "tags": {}
    }
}
```

Configurazione EKS Pod Identity Agent

Disabilita **IPv6** nell'EKS Pod Identity Agent

AWS Management Console

Disabilita **IPv6** in AWS Management Console

1. Per disabilitarlo `IPv6` in EKS Pod Identity Agent, aggiungi la seguente configurazione alle impostazioni di configurazione opzionali del componente aggiuntivo EKS.

   a. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.

b. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui configurare il componente aggiuntivo.

c. Seleziona la scheda Componenti aggiuntivi.

d. Seleziona la casella in alto a destra del componente aggiuntivo EKS Pod Identity Agent, quindi scegli Modifica.

e. Nella pagina Configura EKS Pod Identity Agent:

   i. seleziona la Version (Versione) da utilizzare. Ti consigliamo di mantenere la stessa versione del passaggio precedente e di aggiornare la versione e la configurazione con azioni separate.

   ii. Scegli Impostazioni di configurazione facoltative.

   iii. Inserisci la chiave JSON **`"agent":`** e il valore di un oggetto JSON annidato con una chiave **`"additionalArgs":`** in Valori di configurazione. Il testo risultante deve essere un oggetto JSON valido. Se questa chiave e questo valore sono gli unici dati nella casella di testo, racchiudi la chiave e il valore tra parentesi graffe `{}`. L'esempio seguente mostra che la politica di rete è abilitata:

```
{
    "agent": {
        "additionalArgs": {
            "-b": "169.254.170.23"
        }
    }
}
```

   Questa configurazione imposta l'`IPv4`indirizzo come unico indirizzo utilizzato dall'agente.

f. Per applicare la nuova configurazione sostituendo i pod EKS Pod Identity Agent, scegli Salva modifiche.

   Amazon EKS applica le modifiche ai componenti aggiuntivi EKS utilizzando un'implementazione di EKS Pod Identity Agent. Kubernetes `DaemonSet` Puoi tenere traccia dello stato dell'implementazione nella cronologia degli aggiornamenti del componente aggiuntivo in e con. AWS Management Console `kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system`

   `kubectl rollout`ha i seguenti comandi:

```
$ kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Se l'implementazione richiede troppo tempo, Amazon EKS annullerà l'implementazione e un messaggio con il tipo di Addon Update e lo stato Failed verrà aggiunto alla cronologia degli aggiornamenti del componente aggiuntivo. Per esaminare eventuali problemi, inizia dalla cronologia dell'implementazione ed esegui `kubectl logs` su un pod EKS Pod Identity Agent per visualizzare i log di EKS Pod Identity Agent.

2. Se la nuova voce nella cronologia degli aggiornamenti ha lo stato Riuscito, l'implementazione è stata completata e il componente aggiuntivo utilizza la nuova configurazione in tutti i pod EKS Pod Identity Agent.

AWS CLI

Disabilita in **IPv6**AWS CLI

- Per disabilitarlo `IPv6` in EKS Pod Identity Agent, aggiungi la seguente configurazione ai valori di configurazione del componente aggiuntivo EKS.

  Eseguite il AWS CLI comando seguente. Sostituisci `my-cluster` con il nome del cluster e l'ARN del ruolo IAM con il ruolo che stai utilizzando.

```
aws eks update-addon --cluster-name my-cluster --addon-name eks-pod-identity-
agent \
    --resolve-conflicts PRESERVE --configuration-values '{"agent":
{"additionalArgs": { "-b": "169.254.170.23"}}}'
```

  Questa configurazione imposta l'`IPv4`indirizzo come unico indirizzo utilizzato dall'agente.

Amazon EKS applica le modifiche ai componenti aggiuntivi EKS utilizzando un'implementazione di EKS Pod Identity Agent. Kubernetes DaemonSet Puoi tenere traccia dello stato dell'implementazione nella cronologia degli aggiornamenti del componente aggiuntivo in e con. AWS Management Console `kubectl rollout status daemonset/eks-pod-identity-agent --namespace kube-system`

`kubectl rollout`tha i seguenti comandi:

```
kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Se l'implementazione richiede troppo tempo, Amazon EKS annullerà l'implementazione e un messaggio con il tipo di Addon Update e lo stato Failed verrà aggiunto alla cronologia degli aggiornamenti del componente aggiuntivo. Per esaminare eventuali problemi, inizia dalla cronologia dell'implementazione ed esegui `kubectl logs` su un pod EKS Pod Identity Agent per visualizzare i log di EKS Pod Identity Agent.

Configura un account Kubernetes di servizio per assumere un ruolo IAM con EKS Pod Identity

Questo argomento spiega come configurare un account di Kubernetes servizio per assumere un ruolo AWS Identity and Access Management (IAM) con EKS Pod Identity. Tutti coloro Pods che sono configurati per utilizzare l'account di servizio possono quindi accedere a qualsiasi account a Servizio AWS cui il ruolo dispone delle autorizzazioni di accesso.

Per creare un'associazione EKS Pod Identity, è sufficiente un solo passaggio: si crea l'associazione in EKS tramite AWS Management Console, AWS CLI, AWS SDK AWS CloudFormation e altri strumenti. Non sono presenti dati o metadati sulle associazioni all'interno del cluster in nessun oggetto Kubernetes e non si aggiungono annotazioni agli account di servizio.

Prerequisiti

- Un cluster esistente. Se non se ne possiede già uno, crearlo seguendo una delle guide Guida introduttiva ad Amazon EKS.
- Il principale IAM che sta creando l'associazione deve avere `iam:PassRole`.
- La versione più recente di quella AWS CLI installata e configurata sul dispositivo o AWS CloudShell. È possibile verificare la versione corrente con `aws --version | cut -d / -f2 | cut -d ' ' -f1`. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta Installazione, aggiornamento e disinstallazione AWS CLI e Configurazione rapida `aws configure` nella Guida per l' AWS Command Line Interface utente. La AWS CLI versione installata in AWS CloudShell può anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l' AWS CloudShell utente.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Un file `kubectl config` esistente che contiene la configurazione del cluster. Per creare un file `kubectl config`, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.

Creazione dell'associazione EKS Pod Identity

AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui configurare il componente aggiuntivo di EKS Pod Identity Agent.
3. Scegli la scheda Accesso.
4. In Associazioni Pod Identity, scegli Crea.
5. Per Ruolo IAM, seleziona il ruolo IAM con le autorizzazioni che desideri assegnare al carico di lavoro.

> **Note**
>
> L'elenco contiene solo i ruoli con la seguente policy di attendibilità che consente a EKS Pod Identity di utilizzarli.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

`sts:AssumeRole`

EKS Pod Identity utilizza `AssumeRole` per assumere il ruolo IAM prima di passare le credenziali temporanee ai pod.

`sts:TagSession`

EKS Pod Identity utilizza `TagSession` per includere i tag di sessione nelle richieste a AWS STS.

È possibile utilizzare questi tag in condition keys nella policy di attendibilità per limitare gli account di servizio, gli spazi dei nomi e i cluster che possono utilizzare questo ruolo.

Per un elenco di chiavi di condizione di Amazon EKS, consultare Condizioni per Amazon Elastic Kubernetes Service in Service Authorization Reference. Per informazioni su operazioni e risorse con cui è possibile utilizzare una chiave di condizione, consultare Operazioni definite da Amazon Elastic Kubernetes Service.

6. Per Spazio dei nomi di Kubernetes, seleziona lo spazio dei nomi Kubernetes che contiene l'account di servizio e il carico di lavoro. Facoltativamente, puoi specificare uno spazio dei nomi che non esiste nel cluster.
7. Per Account di servizio Kubernetes, seleziona l'account di servizio Kubernetes da utilizzare. Il manifesto del carico di lavoro Kubernetes deve specificare questo account di servizio. Facoltativamente, puoi specificare un account di servizio in base al nome che non è presente nel cluster.
8. (Facoltativo) Per Tag, scegli Aggiungi tag per aggiungere metadati in una coppia chiave-valore. Questi tag vengono applicati all'associazione e possono essere utilizzati nelle policy IAM.

   Puoi ripetere questo passaggio per aggiungere più tag.
9. Scegli Crea.

AWS CLI

1. Se desideri associare una policy IAM esistente al tuo ruolo IAM, passa al prossimo passaggio.

   Creare una policy IAM Puoi creare una policy personalizzata o copiare una policy AWS gestita che concede già alcune delle autorizzazioni necessarie e personalizzarla a seconda dei requisiti specifici desiderati. Per ulteriori informazioni, consulta Creazione di policy IAM nella Guida per l'utente di IAM.

   a. Crea un file che includa le autorizzazioni per i Servizi AWS a cui vuoi che i tuoi Pods accedano. Per un elenco di tutte le azioni per tutti Servizi AWS, consulta il Service Authorization Reference.

      Puoi eseguire il comando seguente per creare un file policy di esempio che consenta l'accesso in sola lettura a un bucket Amazon S3. Puoi facoltativamente archiviare le informazioni di configurazione o uno script di bootstrap in questo bucket; i container nel Pod possono leggere il file dal bucket e caricarlo nell'applicazione. Se desideri creare questa policy di esempio, copia i seguenti contenuti sul dispositivo. Sostituisci *my-pod-secrets-bucket* con il nome del bucket ed esegui il comando.

      ```
      cat >my-policy.json <<EOF
      {
          "Version": "2012-10-17",
      ```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
EOF
```

b. Creare la policy IAM.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

2. Crea un ruolo IAM e associalo a un account del servizio Kubernetes.

1. Se disponi di un account di servizio Kubernetes esistente che desideri assuma un ruolo IAM, puoi saltare questa fase.

   Crea un account di servizio Kubernetes. Copia i seguenti contenuti sul dispositivo. Sostituisci *my-service-account* con il nome desiderato e *default* con uno spazio dei nomi diverso, se necessario. Se si cambia *default*, lo spazio dei nomi deve già esistere.

```
cat >my-service-account.yaml <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: my-service-account
  namespace: default
EOF
kubectl apply -f my-service-account.yaml
```

   Esegui il comando seguente.

```
kubectl apply -f my-service-account.yaml
```

2. Per creare una policy di attendibilità del ruolo IAM, esegui il comando seguente.

```
cat >trust-relationship.json <<EOF
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
EOF
```

3. Crea il ruolo. Sostituisci *my-role* con un nome per il ruolo IAM e *my-role-description* con una descrizione per il tuo ruolo.

```
aws iam create-role --role-name my-role --assume-role-policy-document
 file://trust-relationship.json --description "my-role-description"
```

4. Allegare una policy IAM al ruolo. Sostituisci *my-role* con il nome del ruolo IAM e *my-policy* con il nome di una policy esistente che hai creato.

```
aws iam attach-role-policy --role-name my-role --policy-
arn=arn:aws:iam::111122223333:policy/my-policy
```

> **Note**
>
> A differenza dei ruoli IAM per gli account di servizio, EKS Pod Identity non utilizza l'annotazione sull'account di servizio.

5. Esegui il comando seguente per creare l'associazione. Sostituisci `my-cluster` con il nome del cluster, *my-service-account* con il nome desiderato e *default* con uno spazio dei nomi diverso, se necessario.

```
aws eks create-pod-identity-association --cluster-name my-cluster --role-
arn arn:aws:iam::111122223333:role/my-role --namespace default --service-
account my-service-account
```

Di seguito viene riportato un output di esempio:

```
{
    "association": {
        "clusterName": "my-cluster",
        "namespace": "default",
        "serviceAccount": "my-service-account",
        "roleArn": "arn:aws:iam::111122223333:role/my-role",
        "associationArn": "arn:aws::111122223333:podidentityassociation/my-
cluster/a-abcdefghijklmnop1",
        "associationId": "a-abcdefghijklmnop1",
        "tags": {},
        "createdAt": 1700862734.922,
        "modifiedAt": 1700862734.922
    }
}
```

> **Note**
>
> Puoi specificare uno spazio dei nomi e un account di servizio in base al nome che non esiste nel cluster. È necessario creare lo spazio dei nomi, l'account di servizio e il carico di lavoro che utilizza l'account di servizio affinché l'associazione EKS Pod Identity funzioni.

3. Conferma che il ruolo e l'account del servizio siano configurati correttamente.

   a. Conferma che la policy di attendibilità del ruolo IAM sia configurata correttamente.

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

   Di seguito viene riportato un output di esempio:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Allow EKS Auth service to assume this role for Pod
 Identities",
            "Effect": "Allow",
            "Principal": {
```

```
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

b. Conferma che la policy che hai associato al tuo ruolo in un passaggio precedente sia associata al ruolo.

```
aws iam list-attached-role-policies --role-name my-role --query
 AttachedPolicies[].PolicyArn --output text
```

Di seguito viene riportato un output di esempio:

```
arn:aws:iam::111122223333:policy/my-policy
```

c. Imposta una variabile per memorizzare il nome della risorsa Amazon (ARN) della policy che desideri utilizzare. Sostituisci *my-policy* con il nome della policy per la quale desideri confermare le autorizzazioni.

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

d. Visualizza la versione predefinita della policy.

```
aws iam get-policy --policy-arn $policy_arn
```

Di seguito viene riportato un output di esempio:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
```

```
}
```

e. Visualizza il contenuto della policy per assicurarti che la policy includa tutte le autorizzazioni di cui Pod ha bisogno. Se necessario, sostituisci *1* nel comando seguente con la versione restituita nell'output della fase precedente.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

Di seguito viene riportato un output di esempio:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

Se hai creato la policy di esempio in un passaggio precedente, il risultato è lo stesso. Se hai creato una policy diversa, allora il contenuto dell'*esempio* è diverso.

Approfondimenti

PodsConfigurare l'utilizzo di un account Kubernetes di servizio

PodsConfigurare l'utilizzo di un account Kubernetes di servizio

Se è Pod necessario accedere Servizi AWS, è necessario configurarlo per utilizzare un account Kubernetes di servizio. L'account di servizio deve essere associato a un ruolo AWS Identity and Access Management (IAM) che dispone delle autorizzazioni per accedere a. Servizi AWS

Prerequisiti

- Un cluster esistente. Se non se ne possiede già uno, crearlo utilizzando una delle guide Guida introduttiva ad Amazon EKS.
- Un account di servizio Kubernetes esistente e un'associazione EKS Pod Identity che associa l'account di servizio a un ruolo IAM. Il ruolo deve avere una policy IAM associata che contenga

le autorizzazioni che desideri che i tuoi Pods posseggano per utilizzare Servizi AWS. Per ulteriori informazioni su come creare e configurare l'account e il ruolo del servizio, consulta Configura un account Kubernetes di servizio per assumere un ruolo IAM con EKS Pod Identity.

- La versione più recente di quella AWS CLI installata e configurata sul dispositivo o AWS CloudShell. È possibile verificare la versione corrente con `aws --version | cut -d / -f2 | cut -d ' ' -f1`. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta Installazione, aggiornamento e disinstallazione AWS CLI e Configurazione rapida `aws configure` nella Guida per l' AWS Command Line Interface utente. La AWS CLI versione installata in AWS CloudShell può anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l' AWS CloudShell utente.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Un file `kubectl config` esistente che contiene la configurazione del cluster. Per creare un file `kubectl config`, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.

Configurazione di un Pod per l'utilizzo di un account di servizio

1. Utilizza il seguente comando per creare un manifesto di implementazione da implementare in un Pod con cui confermare la configurazione. Sostituisci i *`example values`* con i valori in tuo possesso.

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
```

```
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. Implementa il file manifesto al cluster.

```
kubectl apply -f my-deployment.yaml
```

3. Verifica che le variabili di ambiente richieste esistano per Pod.

   a. Visualizzare i Pods distribuiti con l'implementazione nella fase precedente.

   ```
   kubectl get pods | grep my-app
   ```

   Di seguito viene riportato un output di esempio:

   ```
   my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
   ```

   b. Verifica che il Pod abbia il file di token dell'account di servizio montato.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
    AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:
   ```

   Di seguito viene riportato un output di esempio:

   ```
   AWS_CONTAINER_AUTHORIZATION_TOKEN_FILE:  /var/run/secrets/
   pods.eks.amazonaws.com/serviceaccount/eks-pod-identity-token
   ```

4. Conferma di Pods poter interagire con l' Servizi AWS utilizzo delle autorizzazioni assegnate nella policy IAM allegata al tuo ruolo.

> **Note**
>
> Quando un Pod utilizza AWS le credenziali di un ruolo IAM associato a un account di servizio, gli SDK AWS CLI o altri SDK nei relativi contenitori Pod utilizzano le credenziali fornite da quel ruolo. Se non si limita l'accesso alle credenziali fornite al ruolo IAM

del nodo Amazon EKS, il Pod ha comunque accesso a tali credenziali. Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.

Se il tuo Pods non riesce a interagire con i servizi come previsto, completa i seguenti passaggi per confermare che tutto sia configurato correttamente.

a. Conferma di Pods utilizzare una versione AWS SDK che supporti l'assunzione di un ruolo IAM tramite un'associazione EKS Pod Identity. Per ulteriori informazioni, consulta Usa un AWS SDK supportato.
b. Conferma che l'implementazione stia utilizzando l'account del servizio.

```
kubectl describe deployment my-app | grep "Service Account"
```

Di seguito viene riportato un output di esempio:

```
Service Account:  my-service-account
```

Definire le autorizzazioni per fare in modo che le associazioni EKS Pod Identity assumano i ruoli in base ai tag

EKS Pod Identity allega i tag alle credenziali temporanee di ciascun pod con attributi come nome cluster, spazio dei nomi, nome account di servizio. Questi tag di sessione di ruolo consentono agli amministratori di creare un singolo ruolo che può funzionare su più account di servizio, consentendo l'accesso alle AWS risorse in base ai tag corrispondenti. Aggiungendo il supporto per i tag di sessione dei ruoli, i clienti possono imporre limiti di sicurezza più rigorosi tra i cluster e i carichi di lavoro all'interno dei cluster, riutilizzando al contempo gli stessi ruoli IAM e le stesse policy IAM.

Ad esempio, la policy di seguito consente l'azione `s3:GetObject` se l'oggetto è contrassegnato con il nome del cluster EKS.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "s3:ListAllMyBuckets"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:GetObjectTagging"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "s3:ExistingObjectTag/eks-cluster-name": "${aws:PrincipalTag/eks-
cluster-name}"
                }
            }
        }
    ]
}
```

Elenco di tag di sessione aggiunti da EKS Pod Identity

L'elenco seguente contiene tutte le chiavi per i tag che vengono aggiunte alla richiesta `AssumeRole` effettuata da Amazon EKS. Per utilizzare questi tag nelle policy, usa `${aws:PrincipalTag/` seguito dalla chiave, ad esempio `${aws:PrincipalTag/kubernetes-namespace}`.

- `eks-cluster-arn`
- `eks-cluster-name`
- `kubernetes-namespace`
- `kubernetes-service-account`
- `kubernetes-pod-name`
- `kubernetes-pod-uid`

Tag tra account

Tutti i tag di sessione aggiunti da EKS Pod Identity sono transitivi; le chiavi e i valori dei tag vengono passati a tutte le azioni `AssumeRole` utilizzate dai carichi di lavoro per cambiare ruolo in un altro account. È possibile utilizzare questi tag nelle policy di altri account per limitare l'accesso in scenari

tra account. Per ulteriori informazioni, consulta Concatenamento di ruoli con i tag di sessione nella Guida per l'utente di IAM.

Tag personalizzati

EKS Pod Identity non può aggiungere tag personalizzati supplementari all'azione `AssumeRole` che esegue. Tuttavia, i tag applicati al ruolo IAM sono sempre disponibili nello stesso formato: `${aws:PrincipalTag/` seguito dalla chiave, ad esempio `${aws:PrincipalTag/MyCustomTag}`.

> **Note**
>
> I tag aggiunti alla sessione tramite la richiesta `sts:AssumeRole` hanno la precedenza in caso di conflitto. Ad esempio, supponiamo che Amazon EKS aggiunga una chiave `eks-cluster-name` e un valore `my-cluster` alla sessione quando EKS assume il ruolo di cliente. Hai aggiunto anche un tag `eks-cluster-name` al ruolo IAM con valore `my-own-cluster`. In questo caso, il primo ha la precedenza e il valore del tag `eks-cluster-name` sarà `my-cluster`.

Usa un AWS SDK supportato

> **Important**
>
> Una versione precedente della documentazione non era corretta. L' AWS SDK for Java v1 non supporta EKS Pod Identity.

Durante l'utilizzoEKS Pod Identity, i contenitori presenti Pods devono utilizzare una versione AWS SDK che supporti l'assunzione di un ruolo IAM da parte dell'EKS Pod Identity Agent. Assicurati di utilizzare le seguenti versioni, o successive, per il tuo AWS SDK:

- Java (Versione 2) – 2.21.30
- Vai a v1 – v1.47.11
- Vai a v2 – release-2023-11-14
- Python (Boto3) — 1.34.41
- Python (botocore) — 1.34.41
- AWS CLI — 1.30.0

AWS CLI — 2.15.0

- JavaScript v2 — 2.1550.0
- JavaScript v3 — v3.458.0
- Kotlin — versione 1.0.1
- Ruby – 3.188.0
- Rust — rilascio-2024-03-13
- C++ — 1.11.263
- .NET — 3.7.734.0
- PowerShell — 4.1.502
- PHP – 3.287.1

Per assicurarti che l'SDK utilizzato sia supportato, quando crei i container segui la procedura di installazione per il tuo SDK preferito disponibile in Strumenti per costruire su AWS.

Per un elenco di componenti aggiuntivi che supportano EKS Pod Identity, vedi. Versioni aggiuntive compatibili con EKS Pod Identity

Utilizzo delle credenziali EKS Pod Identity

Per utilizzare le credenziali di un'associazione EKS Pod Identity, il codice può utilizzare qualsiasi AWS SDK per creare un client per un AWS servizio con un SDK e, per impostazione predefinita, l'SDK cerca le credenziali da utilizzare in una catena di posizioni. AWS Identity and Access Management Si utilizzano le credenziali EKS Pod Identity se non si specifica un provider di credenziali quando si crea il client o si inizializza in altro modo l'SDK.

Questo avviene perché le associazioni EKS Pod Identity sono state aggiunte al provider di credenziali del container, che viene cercato in un passaggio nella catena di credenziali predefinita. Se i carichi di lavoro utilizzano attualmente credenziali che si trovano all'inizio della catena di credenziali, quest'ultime continuano a essere utilizzate anche se configuri un'associazione EKS Pod Identity per lo stesso carico di lavoro.

Per ulteriori informazioni sul funzionamento delle associazioni EKS Pod Identity, consulta Come funziona EKS Pod Identity.

Ruolo di EKS Pod Identity

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowEksAuthToAssumeRoleForPodIdentity",
            "Effect": "Allow",
            "Principal": {
                "Service": "pods.eks.amazonaws.com"
            },
            "Action": [
                "sts:AssumeRole",
                "sts:TagSession"
            ]
        }
    ]
}
```

`sts:AssumeRole`

EKS Pod Identity utilizza `AssumeRole` per assumere il ruolo IAM prima di passare le credenziali temporanee ai pod.

`sts:TagSession`

EKS Pod Identity utilizza `TagSession` per includere i tag di sessione nelle richieste a AWS STS.

È possibile utilizzare questi tag in condition keys nella policy di attendibilità per limitare gli account di servizio, gli spazi dei nomi e i cluster che possono utilizzare questo ruolo.

Per un elenco di chiavi di condizione di Amazon EKS, consultare Condizioni per Amazon Elastic Kubernetes Service in Service Authorization Reference. Per informazioni su operazioni e risorse con cui è possibile utilizzare una chiave di condizione, consultare Operazioni definite da Amazon Elastic Kubernetes Service.

Ruoli IAM per gli account di servizio

Le applicazioni nei contenitori Pod a's possono utilizzare un AWS SDK o AWS CLI effettuare richieste API per Servizi AWS utilizzare le autorizzazioni AWS Identity and Access Management (IAM). Le applicazioni devono firmare le proprie richieste AWS API con AWS credenziali. I ruoli IAM per gli account di servizio forniscono la possibilità di gestire le credenziali per le applicazioni, analogamente al modo in cui i profili di istanza Amazon EC2 forniscono le credenziali alle istanze Amazon EC2. Invece di creare e distribuire AWS le tue credenziali nei contenitori o utilizzare il ruolo dell'istanza

Amazon EC2, associ un ruolo IAM a un account di servizio e configuri Kubernetes Pods il tuo per l'utilizzo dell'account di servizio. Non è possibile utilizzare i ruoli IAM per gli account di servizio con i cluster locali per Amazon EKS su AWS Outposts.

La caratteristica dei ruoli IAM per gli account di servizio offre i seguenti vantaggi:

- Privilegio minimo: è possibile definire l'ambito delle autorizzazioni IAM per un account di servizio; solo i Pods che utilizzano tale account avranno accesso alle autorizzazioni definite. Questa caratteristica elimina anche la necessità di soluzioni di terze parti, ad esempio `kiam` o `kube2iam`.
- Isolamento delle credenziali: un container del Pod's può recuperare solo le credenziali per il ruolo IAM associato all'account del servizio che utilizza. Un container non ha mai accesso alle credenziali utilizzate da altri container in Pods. Quando si utilizzano ruoli IAM per gli account di servizio, i container Pod's hanno anche le autorizzazioni assegnate al nodo del ruolo IAM Amazon EKS, a meno che non blocchi l'accesso Pod al servizio di metadati di istanza (IMDS) di Amazon EC2. Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.
- Verificabilità: la registrazione degli accessi e degli eventi è disponibile AWS CloudTrail per garantire un controllo retrospettivo.

Abilita i ruoli IAM per gli account di servizio completando le seguenti procedure:

1. Crea un OIDC provider IAM per il tuo cluster: questa procedura viene completata una sola volta per ogni cluster.

> **Note**
>
> Se abiliti l'endpoint EKS VPC, non è possibile accedere all'endpoint del servizio EKS OIDC dall'interno di quel VPC. Di conseguenza, le operazioni come la creazione di un provider OIDC con `eksctl` nel VPC non funzioneranno e comporteranno un timeout durante il tentativo di richiesta di `https://oidc.eks.region.amazonaws.com`. Segue un messaggio di errore di esempio:
>
> ```
> ** server can't find oidc.eks.region.amazonaws.com: NXDOMAIN
> ```
>
> Per completare questo passaggio, puoi eseguire il comando all'esterno del VPC, ad esempio in AWS CloudShell o su un computer connesso a Internet.

2. Configurare un account Kubernetes di servizio per assumere un ruolo IAM: completa questa procedura per ogni set univoco di autorizzazioni che desideri abbia un'applicazione.
3. PodsConfigurare l'utilizzo di un account Kubernetes di servizio— Completa questa procedura per ogni utente a Pod cui è necessario accedere Servizi AWS.
4. Utilizzo di un SDK AWS supportato— Conferma che il carico di lavoro utilizzi un AWS SDK di una versione supportata e che utilizzi la catena di credenziali predefinita.

IAM, Kubernetes, e OpenID Connect (OIDC) informazioni di base

Nel 2014, è AWS Identity and Access Management stato aggiunto il supporto per le identità federate utilizzando (). OpenID Connect OIDC Questa funzionalità consente di autenticare le chiamate AWS API con i provider di identità supportati e di ricevere un token OIDC JSON web valido (). JWT Puoi passare questo token all'operazione AWS STS `AssumeRoleWithWebIdentity` API e ricevere le credenziali del ruolo temporaneo IAM. È possibile utilizzare queste credenziali per interagire con qualsiasi servizio Servizio AWS, tra cui Amazon S3 e DynamoDB.

Ogni token JWT è firmato da una coppia di chiavi di firma. Le chiavi vengono fornite dal provider OIDC gestito da Amazon EKS e la chiave privata ruota ogni 7 giorni. Amazon EKS conserva le chiavi pubbliche fino alla loro scadenza. Se connetti client OIDC esterni, tieni presente che devi aggiornare le chiavi di firma prima della scadenza della chiave pubblica. Scopri come effettuare il the section called “Recupera le chiavi di firma”.

Kubernetes utilizza da tempo gli account di servizio come sistema di identità interno. Pods può autenticarsi con il Kubernetes Server API che utilizza un token montato automaticamente (che non era OIDC JWT) che solo il Kubernetes server API potrebbe convalidare. Questi token legacy dell'account di servizio non hanno scadenza e la rotazione della chiave di firma è un processo difficile. Nella Kubernetes versione `1.12`, è stato aggiunto il supporto per una nuova `ProjectedServiceAccountToken` caratteristica. Questa funzionalità è un token Web OIDC JSON che contiene anche l'identità dell'account del servizio e supporta un pubblico configurabile.

Amazon EKS ora ospita un endpoint di rilevamento OIDC pubblico per ogni cluster, contenente le chiavi di firma per i token Web `ProjectedServiceAccountToken` JSON, in modo che i sistemi esterni, ad esempio IAM, possano convalidare e accettare i token OIDC emessi da Kubernetes.

Crea un OIDC provider IAM per il tuo cluster

Il cluster ha un emittente URL OpenID Connect (OIDC) associato. Per utilizzare i ruoli AWS Identity and Access Management (IAM) per gli account di servizio, deve esistere un OIDC provider IAM per l'URL OIDC emittente del cluster.

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Un file `kubectl config` esistente che contiene la configurazione del cluster. Per creare un file `kubectl config`, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.

Puoi creare un provider OIDC IAM per il tuo cluster utilizzando `eksctl` o AWS Management Console.

eksctl

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Creazione di un provider di identità OIDC IAM per il cluster con **`eksctl`**

1. Determina l'ID dell'emittente OIDC per il tuo cluster.

   Recupera gli ID dell'emittente OIDC del cluster e archivialo in una variabile. Sostituire *`my-cluster`* con il proprio valore.

   ```
   cluster_name=my-cluster
   ```

   ```
   oidc_id=$(aws eks describe-cluster --name $cluster_name --query
    "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
   ```

   ```
   echo $oidc_id
   ```

2. Determina se un fornitore OIDC IAM con l'ID del tuo cluster è già presente nel tuo account.

   ```
   aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
   ```

   Se viene restituito un output, già disponi di un provider OIDC IAM per il tuo cluster e puoi saltare al passaggio successivo. Se non viene restituito alcun output, devi creare un provider IAM OIDC per il cluster.

3. Creare un provider di identità OIDC IAM per il cluster con il comando seguente.

   ```
   eksctl utils associate-iam-oidc-provider --cluster $cluster_name --approve
   ```

   > **Note**
   >
   > Se abiliti l'endpoint EKS VPC, non è possibile accedere all'endpoint del servizio EKS OIDC dall'interno di quel VPC. Di conseguenza, le operazioni come la creazione di un provider OIDC con `eksctl` nel VPC non funzioneranno e comporteranno un timeout durante il tentativo di richiesta di `https://oidc.eks.`*`region`*`.amazonaws.com`. Segue un messaggio di errore di esempio:

```
** server can't find oidc.eks.region.amazonaws.com: NXDOMAIN
```

Per completare questo passaggio, puoi eseguire il comando all'esterno del VPC, ad esempio in AWS CloudShell o su un computer connesso a Internet.

AWS Management Console

Per creare un provider di OIDC identità IAM per il cluster con AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro a sinistra, seleziona Cluster, quindi seleziona il nome del cluster nella pagina Cluster.
3. Nella sezione Dettagli della scheda Panoramica, annota il valore dell'URL del provider OpenID Connect.
4. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
5. Nel pannello di navigazione, scegli Identity Providers (Provider di identità) in Access management (Gestione accesso). Se un Provider nell'elenco corrisponde all'URL del cluster, significa che si dispone già di un provider per il cluster. Se nell'elenco non è presente un provider che corrisponde all'URL del cluster, è necessario crearne uno.
6. Per creare un provider, selezionare Aggiungi provider.
7. Per Provider type (Tipo di fornitore), seleziona OpenID Connect.
8. Per URL provider, incollare l'URL del provider OIDC per il cluster, quindi scegliere Ottieni l'identificazione personale.
9. Per Destinatari, inserire **`sts.amazonaws.com`** e scegliere Aggiungi provider.

Approfondimenti

Configurare un account Kubernetes di servizio per assumere un ruolo IAM

Configurare un account Kubernetes di servizio per assumere un ruolo IAM

Questo argomento spiega come configurare un account Kubernetes di servizio per assumere un ruolo AWS Identity and Access Management (IAM). Qualsiasi Pods configurato per utilizzare

l'account del servizio può quindi accedere a qualsiasi Servizio AWS per cui il ruolo dispone delle autorizzazioni di accesso.

Prerequisiti

- Un cluster esistente. Se non se ne possiede già uno, crearlo seguendo una delle guide Guida introduttiva ad Amazon EKS.
- Provider IAM OpenID Connect (OIDC) esistente per il cluster. Per sapere se disponi di un account o, per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Un file `kubectl config` esistente che contiene la configurazione del cluster. Per creare un file `kubectl config`, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.

Per associare un ruolo IAM a un account del servizio Kubernetes.

1. Se desideri associare una policy IAM esistente al tuo ruolo IAM, passa al prossimo passaggio.

   Creare una policy IAM È possibile creare una politica personalizzata o copiare una politica AWS gestita che già concede alcune delle autorizzazioni necessarie e personalizzarla in base alle proprie esigenze specifiche. Per ulteriori informazioni, consulta Creazione di policy IAM nella Guida per l'utente di IAM.

a. Crea un file che includa le autorizzazioni per il Servizi AWS quale desideri accederePods. Per un elenco di tutte le azioni per tutti Servizi AWS, consulta il Service Authorization Reference.

   Puoi eseguire il comando seguente per creare un file policy di esempio che consenta l'accesso in sola lettura a un bucket Amazon S3. Puoi facoltativamente archiviare le informazioni di configurazione o uno script di bootstrap in questo bucket; i container nel Pod possono leggere il file dal bucket e caricarlo nell'applicazione. Se desideri creare questa policy di esempio, copia i seguenti contenuti sul dispositivo. Sostituisci *my-pod-secrets-bucket* con il nome del bucket ed esegui il comando.

```
cat >my-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
EOF
```

b. Creare la policy IAM.

```
aws iam create-policy --policy-name my-policy --policy-document file://my-policy.json
```

2. Crea un ruolo IAM e associalo a un account del servizio Kubernetes. In alternativa, è possibile utilizzare `eksctl` o la AWS CLI.

   eksctl

   Prerequisito

   La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Sostituisci *my-service-account* con il nome dell'account del servizio Kubernetes per cui desideri che `eksctl` crei e associ il ruolo IAM. Sostituisci *default* con lo spazio dei nomi in cui desideri che `eksctl` crei l'account del servizio. Sostituisci *my-cluster* con il nome del tuo cluster. Sostituisci *my-role* con il nome del ruolo a cui associare l'account di servizio. Se non esiste già, `eksctl` lo crea per te. Sostituisci *111122223333* con il tuo ID account e *my-policy* con il nome della policy .

```
eksctl create iamserviceaccount --name my-service-account --namespace default --
cluster my-cluster --role-name my-role \
    --attach-policy-arn arn:aws:iam::111122223333:policy/my-policy --approve
```

> **Important**
>
> Se il ruolo o l'account del servizio esiste già, il comando precedente potrebbe non riuscire. `eksctl` ha diverse opzioni che puoi fornire in queste situazioni. Per ulteriori informazioni, esegui **`eksctl create iamserviceaccount --help`**.

AWS CLI

1. Se disponi di un account di servizio Kubernetes esistente che desideri assuma un ruolo IAM, puoi saltare questa fase.

   Crea un account di servizio Kubernetes. Copia i seguenti contenuti sul dispositivo. Sostituisci *my-service-account* con il nome desiderato e *default* con uno spazio dei nomi diverso, se necessario. Se si cambia *default*, lo spazio dei nomi deve già esistere.

   ```
   cat >my-service-account.yaml <<EOF
   apiVersion: v1
   kind: ServiceAccount
   metadata:
     name: my-service-account
     namespace: default
   EOF
   kubectl apply -f my-service-account.yaml
   ```

2. Imposta il tuo Account AWS ID su una variabile di ambiente con il seguente comando.

```
account_id=$(aws sts get-caller-identity --query "Account" --output text)
```

3. Impostare il provider di identità OIDC del cluster su una variabile di ambiente con il comando seguente. Sostituisci *my-cluster* con il nome del cluster.

```
oidc_provider=$(aws eks describe-cluster --name my-cluster --region
 $AWS_REGION --query "cluster.identity.oidc.issuer" --output text | sed -e "s/
^https:\/\///")
```

4. Imposta le variabili per lo spazio dei nomi e il nome dell'account del servizio. Sostituisci *my-service-account* con l'account di servizio Kubernetes che desideri assuma il ruolo. Sostituisci *default* con lo spazio dei nomi dell'account del servizio.

```
export namespace=default
export service_account=my-service-account
```

5. Per creare una policy di attendibilità del ruolo IAM, esegui il comando seguente. Se desideri consentire a tutti gli account di servizio all'interno di uno spazio dei nomi di utilizzare il ruolo, copia i seguenti contenuti sul tuo dispositivo. Sostituisci *StringEquals*con **StringLike** e sostituisci *$service_account* con. * È possibile aggiungere più voci nelle condizioni `StringEquals` e `StringLike` riportate di seguito per utilizzare più account di servizio o spazi dei nomi con il ruolo. Per consentire a ruoli di un diverso Account AWS rispetto all'account in cui si trova il cluster di assumere il ruolo, consulta Autorizzazioni multi-account IAM per ulteriori informazioni.

```
cat >trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::$account_id:oidc-provider/$oidc_provider"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "$oidc_provider:aud": "sts.amazonaws.com",
          "$oidc_provider:sub": "system:serviceaccount:
$namespace:$service_account"
```

```
        }
      }
    }
  ]
}
EOF
```

6. Crea il ruolo. Sostituisci *my-role* con un nome per il ruolo IAM e *my-role-description* con una descrizione per il tuo ruolo.

```
aws iam create-role --role-name my-role --assume-role-policy-document
 file://trust-relationship.json --description "my-role-description"
```

7. Allegare una policy IAM al ruolo. Sostituisci *my-role* con il nome del ruolo IAM e *my-policy* con il nome di una policy esistente che hai creato.

```
aws iam attach-role-policy --role-name my-role --policy-arn=arn:aws:iam::
$account_id:policy/my-policy
```

8. Annota l'account del servizio con il nome della risorsa Amazon (ARN) del ruolo IAM che desideri che l'account del servizio assuma. Sostituisci *my-role* con il nome del ruolo IAM esistente. Supponiamo di aver consentito a un ruolo di un account Account AWS diverso da quello in cui si trova il cluster di assumere il ruolo in un passaggio precedente. Quindi, assicurati di specificare il ruolo Account AWS and dell'altro account. Per ulteriori informazioni, consulta Autorizzazioni multi-account IAM.

```
kubectl annotate serviceaccount -n $namespace $service_account
 eks.amazonaws.com/role-arn=arn:aws:iam::$account_id:role/my-role
```

3. Conferma che il ruolo e l'account del servizio siano configurati correttamente.

   a. Conferma che la policy di attendibilità del ruolo IAM sia configurata correttamente.

```
aws iam get-role --role-name my-role --query Role.AssumeRolePolicyDocument
```

   Di seguito viene riportato un output di esempio:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:default:my-
service-account",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
                }
            }
        }
    ]
}
```

b. Conferma che la policy che hai associato al tuo ruolo in un passaggio precedente sia associata al ruolo.

```
aws iam list-attached-role-policies --role-name my-role --query
 AttachedPolicies[].PolicyArn --output text
```

Di seguito viene riportato un output di esempio:

```
arn:aws:iam::111122223333:policy/my-policy
```

c. Imposta una variabile per memorizzare il nome della risorsa Amazon (ARN) della policy che desideri utilizzare. Sostituisci *my-policy* con il nome della policy per la quale desideri confermare le autorizzazioni.

```
export policy_arn=arn:aws:iam::111122223333:policy/my-policy
```

d. Visualizza la versione predefinita della policy.

```
aws iam get-policy --policy-arn $policy_arn
```

Di seguito viene riportato un output di esempio:

```
{
    "Policy": {
        "PolicyName": "my-policy",
        "PolicyId": "EXAMPLEBIOWGLDEXAMPLE",
        "Arn": "arn:aws:iam::111122223333:policy/my-policy",
        "Path": "/",
        "DefaultVersionId": "v1",
        [...]
    }
}
```

e. Visualizza il contenuto della policy per assicurarti che la policy includa tutte le autorizzazioni di cui Pod ha bisogno. Se necessario, sostituisci *1* nel comando seguente con la versione restituita nell'output della fase precedente.

```
aws iam get-policy-version --policy-arn $policy_arn --version-id v1
```

Di seguito viene riportato un output di esempio:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::my-pod-secrets-bucket"
        }
    ]
}
```

Se hai creato la policy di esempio in un passaggio precedente, il risultato è lo stesso. Se hai creato una policy diversa, allora il contenuto dell'*esempio* è diverso.

f. Conferma che l'account del servizio Kubernetes sia annotato con il ruolo.

```
kubectl describe serviceaccount my-service-account -n default
```

Di seguito viene riportato un output di esempio:

```
Name:                my-service-account
```

```
Namespace:           default
Annotations:         eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/my-role
Image pull secrets:  <none>
Mountable secrets:   my-service-account-token-qqjfl
Tokens:              my-service-account-token-qqjfl
[...]
```

4. (Facoltativo)Configurare l' AWS Security Token Service endpoint per un account di servizio. AWS consiglia di utilizzare un AWS STS endpoint regionale anziché un endpoint globale. Ciò riduce la latenza, fornisce una ridondanza integrata e aumenta la validità del token di sessione.

Approfondimenti

PodsConfigurare l'utilizzo di un account Kubernetes di servizio

PodsConfigurare l'utilizzo di un account Kubernetes di servizio

Se è Pod necessario accedere Servizi AWS, è necessario configurarlo per utilizzare un account Kubernetes di servizio. L'account di servizio deve essere associato a un ruolo AWS Identity and Access Management (IAM) che dispone delle autorizzazioni per accedere a. Servizi AWS

Prerequisiti

- Un cluster esistente. Se non se ne possiede già uno, crearlo utilizzando una delle guide Guida introduttiva ad Amazon EKS.
- Provider IAM OpenID Connect (OIDC) esistente per il cluster. Per sapere se disponi di un account o, per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- Un account di servizio Kubernetes esistente associato a un ruolo IAM. L'account del servizio deve essere annotato con il nome della risorsa Amazon (ARN) del ruolo IAM. Il ruolo deve avere una policy IAM associata che contenga le autorizzazioni che desideri che i tuoi Pods posseggano per utilizzare Servizi AWS. Per ulteriori informazioni su come creare e configurare l'account e il ruolo del servizio, consulta Configurare un account Kubernetes di servizio per assumere un ruolo IAM.
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws

configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.

- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Un file `kubectl config` esistente che contiene la configurazione del cluster. Per creare un file `kubectl config`, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.

Configurazione di un Pod per l'utilizzo di un account di servizio

1. Utilizza il seguente comando per creare un manifesto di implementazione da implementare in un Pod con cui confermare la configurazione. Sostituisci i *`example values`* con i valori in tuo possesso.

```
cat >my-deployment.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app
spec:
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
    spec:
      serviceAccountName: my-service-account
      containers:
      - name: my-app
        image: public.ecr.aws/nginx/nginx:X.XX
EOF
```

2. Implementa il file manifesto al cluster.

```
kubectl apply -f my-deployment.yaml
```

3. Verifica che le variabili di ambiente richieste esistano per Pod.

   a. Visualizzare i Pods distribuiti con l'implementazione nella fase precedente.

   ```
   kubectl get pods | grep my-app
   ```

   Di seguito viene riportato un output di esempio:

   ```
   my-app-6f4dfff6cb-76cv9   1/1     Running   0          3m28s
   ```

   b. Visualizzare l'ARN del ruolo IAM che Pod sta utilizzando.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep AWS_ROLE_ARN:
   ```

   Di seguito viene riportato un output di esempio:

   ```
   AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/my-role
   ```

   Il ruolo ARN deve corrispondere al ruolo ARN con cui hai annotato l'account del servizio esistente. Per ulteriori informazioni sull'annotazione dell'account del servizio, consulta Configurare un account Kubernetes di servizio per assumere un ruolo IAM.

   c. Conferma che Pod abbia un file di token di identità web montato.

   ```
   kubectl describe pod my-app-6f4dfff6cb-76cv9 | grep
    AWS_WEB_IDENTITY_TOKEN_FILE:
   ```

   Di seguito viene riportato un output di esempio:

   ```
   AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
   serviceaccount/token
   ```

   `kubelet` richiede e archivia il token per conto del Pod. Per impostazione predefinita, la `kubelet` aggiorna il token se è più vecchio dell'80% del suo tempo totale di vita o se il token è più vecchio di 24 ore. Puoi modificare la durata di scadenza per qualsiasi account, ad eccezione dell'account di servizio predefinito, con le impostazioni nelle specifiche del

Pod. Per ulteriori informazioni, consulta Proiezione dei volumi di token dell'account di servizio nella documentazione di Kubernetes.

Il Webhook Amazon EKS Pod Identity sugli orologi del cluster per Pods che utilizzano un account di servizio con la seguente annotazione:

```
eks.amazonaws.com/role-arn: arn:aws:iam::111122223333:role/my-role
```

Il webhook applica le variabili di ambiente precedenti a questi Pods. Il cluster non deve utilizzare il webhook per configurare le variabili di ambiente e i montaggi del file di token. Puoi configurare manualmente Pods per ottenere queste variabili di ambiente. Le versioni supportate dell' AWS SDK cercano queste variabili di ambiente innanzitutto nel provider della catena di credenziali. Le credenziali del ruolo vengono utilizzate per i Pods che soddisfano questi criteri.

4. Conferma di Pods poter interagire con l' Servizi AWS utilizzo delle autorizzazioni assegnate nella policy IAM associata al tuo ruolo.

> **Note**
>
> Quando un Pod utilizza AWS le credenziali di un ruolo IAM associato a un account di servizio, gli SDK AWS CLI o altri SDK nei relativi contenitori Pod utilizzano le credenziali fornite da quel ruolo. Se non si limita l'accesso alle credenziali fornite al ruolo IAM del nodo Amazon EKS, il Pod ha comunque accesso a tali credenziali. Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.

Se il tuo Pods non riesce a interagire con i servizi come previsto, completa i seguenti passaggi per confermare che tutto sia configurato correttamente.

a. Conferma di Pods utilizzare una versione AWS SDK che supporti l'assunzione di un ruolo IAM tramite un OpenID Connect file di token di identità Web. Per ulteriori informazioni, consulta Utilizzo di un SDK AWS supportato.

b. Conferma che l'implementazione stia utilizzando l'account del servizio.

```
kubectl describe deployment my-app | grep "Service Account"
```

Di seguito viene riportato un output di esempio:

```
Service Account:  my-service-account
```

c. Se il tuo Pods non riesce ancora ad accedere ai servizi, controlla i passaggi descritti in Configurare un account Kubernetes di servizio per assumere un ruolo IAM per confermare che il ruolo e l'account del servizio siano configurati correttamente.

Configurare l' AWS Security Token Service endpoint per un account di servizio

Se utilizzi un account di Kubernetes servizio conRuoli IAM per gli account di servizio, puoi configurare il tipo di AWS Security Token Service endpoint utilizzato dall'account di servizio se la versione del cluster e della piattaforma sono uguali o successive a quelle elencate nella tabella seguente. Se le versioni di Kubernetes o della piattaforma sono precedenti a quelle riportate nella tabella, gli account del servizio possono utilizzare esclusivamente l'endpoint globale.

Versione Kubernetes	Versione della piattaforma	Tipo di endpoint predefinito
1.30	eks.2	Regionale
1.29	eks.1	Regionale
1.28	eks.1	Regionale
1.27	eks.1	Regionale
1.26	eks.1	Regionale
1.25	eks.1	Regionale
1.24	eks.2	Regionale
1.23	eks.1	Regionale

AWS consiglia di utilizzare gli AWS STS endpoint regionali anziché l'endpoint globale. Ciò riduce la latenza, fornisce una ridondanza integrata e aumenta la validità del token di sessione. AWS Security Token Service Deve essere attivo nel luogo in Regione AWS cui Pod è in esecuzione. Inoltre, l'applicazione deve avere una ridondanza integrata per un altro Regione AWS in caso di guasto del

servizio nel. Regione AWS Per ulteriori informazioni, consulta Managing AWS STS in an Regione AWS nella IAM User Guide.

Prerequisiti

- Un cluster esistente. Se non se ne possiede già uno, crearlo utilizzando una delle guide Guida introduttiva ad Amazon EKS.
- Provider IAM OIDC esistente per il cluster. Per ulteriori informazioni, consulta Crea un OIDC provider IAM per il tuo cluster.
- Un account del servizio Kubernetes esistente configurato per l'uso con la funzionalità uoli IAM di Amazon EKS per gli account di servizio.

Configurazione del tipo di endpoint utilizzato da un account del servizio Kubernetes

Negli esempi seguenti è utilizzato l'account del servizio Kubernetes `aws-node` utilizzato dal plug-in CNI di Amazon VPC. Puoi sostituire i *`example values`* con gli account del servizio, i Pods, gli spazi dei nomi e altre risorse personalizzate.

1. Seleziona un Pod che utilizza un account di servizio per il quale desideri modificare l'endpoint. Determina Regione AWS in quale Pod ambiente viene eseguito. Sostituisci *`aws-node-6mfgv`* con il nome del Pod e *`kube-system`* con il nome dello spazio dei nomi di Pod.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep Node:
   ```

   Di seguito viene riportato un output di esempio:

   ```
   ip-192-168-79-166.us-west-2/192.168.79.166
   ```

   Regione AWS
2. Determina il tipo di endpoint utilizzato dall'account di servizio del Pod's.

   ```
   kubectl describe pod aws-node-6mfgv -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
   ```

   Di seguito viene riportato un output di esempio:

   ```
   AWS_STS_REGIONAL_ENDPOINTS: regional
   ```

Se l'endpoint attuale è globale, l'output restituisce `global`. Se non viene restituito alcun output, il tipo di endpoint predefinito è in uso e non è stato sovrascritto.

3. Se le versioni del cluster e della piattaforma corrispondono o sono successive a quelle elencate nella tabella, puoi modificare il tipo di endpoint utilizzato dall'account del servizio dal tipo predefinito con un tipo diverso tramite uno dei comandi seguenti. Sostituisci *`aws-node`* con il nome dell'account del servizio e *`kube-system`* con lo spazio dei nomi dell'account del servizio.

   - Se il tipo di endpoint predefinito o attuale è globale e vuoi modificarlo in regionale:

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=true
     ```

     Se usi Ruoli IAM per gli account di servizio per generare URL S3 prefirmati nell'applicazione in esecuzione nei container dei Pods, il formato dell'URL per gli endpoint regionali è simile al seguente:

     ```
     https://bucket.s3.us-west-2.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
     ```

   - Se il tipo di endpoint predefinito o attuale è regionale e vuoi modificarlo in globale:

     ```
     kubectl annotate serviceaccount -n kube-system aws-node eks.amazonaws.com/sts-regional-endpoints=false
     ```

     Se la tua applicazione effettua richieste esplicite agli endpoint AWS STS globali e non sovrascrivi il comportamento predefinito di utilizzo degli endpoint regionali nei cluster Amazon EKS, le richieste falliranno con un errore. Per ulteriori informazioni, consulta I container dei pod riceveranno il seguente errore: `An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`.

     Se usi Ruoli IAM per gli account di servizio per generare URL S3 prefirmati nell'applicazione in esecuzione nei container dei Pods, il formato dell'URL per gli endpoint globali è simile al seguente:

     ```
     https://bucket.s3.amazonaws.com/path?...&X-Amz-Credential=your-access-key-id/date/us-west-2/s3/aws4_request&...
     ```

Se disponi di un'automazione che prevede l'utilizzo dell'URL prefirmato in un determinato formato o se l'applicazione o le dipendenze a valle che utilizzano URL prefirmati hanno aspettative per l' Regione AWS obiettivo, apporta le modifiche necessarie per utilizzare l'endpoint appropriato. AWS STS

4. Elimina e ricrea i Pods esistenti associati all'account di servizio per applicare le variabili di ambiente delle credenziali. Il web hook di modifica non le applica ai Pods già in esecuzione. Puoi sostituire *Pods*, *kube-system* e *-l k8s-app=aws-node* con le informazioni per il Pods per cui hai impostato l'annotazione.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

5. Conferma che tutti i Pods sono stati riavviati.

```
kubectl get Pods -n kube-system -l k8s-app=aws-node
```

6. Visualizza le variabili di ambiente per una delle Pods. Verifica che il valore `AWS_STS_REGIONAL_ENDPOINTS` sia quello che hai impostato in una fase precedente.

```
kubectl describe pod aws-node-kzbtr -n kube-system |grep AWS_STS_REGIONAL_ENDPOINTS
```

Di seguito viene riportato un output di esempio:

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Autorizzazioni multi-account IAM

É possibile configurare le autorizzazioni multi-account IAM creando un provider di identità dal cluster di un altro account o utilizzando operazioni `AssumeRole` concatenate. Negli esempi seguenti, l'Account A possiede un cluster Amazon EKS che supporta i ruoli IAM per gli account di servizio. I Pods che sono in esecuzione sul cluster devono assumere le autorizzazioni IAM dall'Account B.

Example Crea un provider di identità dal cluster di un altro account

Example

In questo esempio, l'account A fornisce all'account B l'URL dell'emittente OpenID Connect (OIDC) dal relativo cluster. L'account B segue le istruzioni in Crea un OIDC provider IAM per il tuo cluster

e Configurare un account Kubernetes di servizio per assumere un ruolo IAM utilizzando l'URL dell'emittente OIDC dal cluster dell'account A. Quindi, un amministratore del cluster annota quindi l'account di servizio nel cluster dell'account A per utilizzare il ruolo dall'account B (*444455556666*).

```
apiVersion: v1
kind: ServiceAccount
metadata:
  annotations:
    eks.amazonaws.com/role-arn: arn:aws:iam::444455556666:role/account-b-role
```

Example Utilizzo di operazioni **`AssumeRole`** concatenate

Example

In questo esempio, l'account B crea una policy IAM con le autorizzazioni da assegnare ai Pods nel cluster dell'account A. L'account B (*444455556666*) allega tale policy a un ruolo IAM con una relazione di attendibilità che concede ad `AssumeRole` le autorizzazioni per l'account A (*111122223333*).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:root"
      },
      "Action": "sts:AssumeRole",
      "Condition": {}
    }
  ]
}
```

L'account A crea un ruolo con una policy di attendibilità in grado di ottenere le credenziali dal provider di identità creato con l'indirizzo dell'emittente OIDC del cluster.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
```

```
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/oidc.eks.region-
code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity"
    }
  ]
}
```

L'account A allega una policy a tale ruolo con le seguenti autorizzazioni per assumere il ruolo creato dall'account B.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::444455556666:role/account-b-role"
        }
    ]
}
```

Il codice dell'applicazione per i Pods che assumono il ruolo dell'account B utilizza due profili: `account_b_role` e `account_a_role`. Il profilo `account_b_role` utilizza il profilo `account_a_role` come origine. Per il AWS CLI, il `~/.aws/config` file è simile al seguente.

```
[profile account_b_role]
source_profile = account_a_role
role_arn=arn:aws:iam::444455556666:role/account-b-role

[profile account_a_role]
web_identity_token_file = /var/run/secrets/eks.amazonaws.com/serviceaccount/token
role_arn=arn:aws:iam::111122223333:role/account-a-role
```

Per specificare profili concatenati per altri AWS SDK, consulta la documentazione dell'SDK che stai utilizzando. Per ulteriori informazioni, consulta Strumenti su cui costruire. AWS

Utilizzo di un SDK AWS supportato

Durante l'utilizzoRuoli IAM per gli account di servizio, i contenitori presenti Pods devono utilizzare una versione AWS SDK che supporti l'assunzione di un ruolo IAM tramite un file di token di identità OpenID Connect Web. Assicurati di utilizzare le seguenti versioni, o successive, per il tuo AWS SDK:

- Java (versione 2) — 2.10.11
- Java — 1.11.704
- Go — 1.23.13
- Python (Boto3) — 1.9.220
- Python (botocore) — 1.12.200
- AWS CLI — 1.16.232
- Nodo: 2,525.0 e 3,27.0
- Ruby — 3.58.0
- C++ — 1.7.174
- .NET: 3,3.659.1 – Devi includere anche `AWSSDK.SecurityToken`.
- PHP — 3.110.7

Molti componenti aggiuntivi Kubernetes più diffusi, ad esempio Cluster Autoscaler, Che cosa è la AWS Load Balancer Controller? e il Amazon VPC CNI plugin for Kubernetes, supportano ruoli IAM per gli account di servizio.

Per assicurarti che l'SDK utilizzato sia supportato, quando crei i container segui la procedura di installazione per il tuo SDK preferito disponibile in Strumenti per costruire su AWS.

Utilizzo delle credenziali

Per utilizzare le credenziali dei ruoli IAM per gli account di servizio, il codice può utilizzare qualsiasi AWS SDK per creare un client per un AWS servizio con un SDK e, per impostazione predefinita, l'SDK cerca le credenziali da utilizzare in una catena di posizioni. AWS Identity and Access Management I ruoli IAM per le credenziali degli account di servizio vengono utilizzati se non specifichi un provider di credenziali quando crei il client o inizializzi in altro modo l'SDK.

Questo perché i ruoli IAM per gli account di servizio sono stati aggiunti come passaggio nella catena di credenziali predefinita. Se i carichi di lavoro attualmente utilizzano credenziali che si trovano

all'inizio della catena di credenziali, tali credenziali continueranno a essere utilizzate anche se configuri ruoli IAM per gli account di servizio per lo stesso carico di lavoro.

L'SDK scambia automaticamente il OIDC token dell'account di servizio con credenziali temporanee utilizzando l'azione. AWS Security Token Service `AssumeRoleWithWebIdentity` Amazon EKS e questa azione SDK continuano a ruotare le credenziali temporanee rinnovandole prima della scadenza.

Recupera le chiavi di firma

Kubernetesquestioni `ProjectedServiceAccountToken` a ciascuna KubernetesService Account. Questo token è un OIDC token, che è inoltre un tipo diJSON web token (JWT). Amazon EKS ospita un OIDC endpoint pubblico per ogni cluster che contiene le chiavi di firma per il token in modo che i sistemi esterni possano convalidarlo.

Per convalidare un`ProjectedServiceAccountToken`, devi recuperare le chiavi di firma OIDC pubbliche, chiamate anche. JSON Web Key Set (JWKS) Usa queste chiavi nella tua applicazione per convalidare il token. Ad esempio, puoi usare la libreria PyJWT Python per convalidare i token usando queste chiavi. Per ulteriori informazioni su, vedere. `ProjectedServiceAccountToken` the section called “IAM, Kubernetes, e OpenID Connect (OIDC) informazioni di base”

Prerequisiti

- Un provider AWS Identity and Access Management (IAM) OpenID Connect (OIDC) esistente per il tuo cluster. Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- AWS CLI— Uno strumento da riga di comando per lavorare con AWS i servizi, incluso Amazon EKS. Per informazioni, consultare Installazione, aggiornamento e disinstallazione di AWS CLI nella Guida per l'utente di AWS Command Line Interface . Dopo aver installato AWS CLI, ti consigliamo di configurarlo anche. Per ulteriori informazioni, consultare Configurazione rapida con `aws configure` nella Guida per l'utente AWS Command Line Interface .

Recupera le chiavi di firma OIDC pubbliche ()AWS CLI

1. Recupera l'OIDCURL per il tuo cluster Amazon EKS utilizzando. AWS CLI

```
$ aws eks describe-cluster --name my-cluster --query 'cluster.identity.oidc.issuer'
"https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE"
```

2. Recupera la chiave di firma pubblica utilizzando curl o uno strumento simile. Il risultato è un JSON Web Key Set (JWKS).

> **Important**
>
> Amazon EKS limita le chiamate verso l'OIDCendpoint. È necessario memorizzare nella cache la chiave di firma pubblica. Rispetta l'`cache-control`intestazione inclusa nella risposta.

> **Important**
>
> Amazon EKS ruota la chiave di OIDC firma ogni sette giorni.

```
$ curl https://oidc.eks.us-west-2.amazonaws.com/id/8EBDXXXX00BAE/keys
{"keys":
[{"kty":"RSA","kid":"2284XXXX4a40","use":"sig","alg":"RS256","n":"wklbXXXXMVfQ","e":"AQAB"}
```

Nodi Amazon EKS

Un nodo Kubernetes è una macchina che esegue applicazioni containerizzate. Ogni nodo include i seguenti componenti:

- Runtime del container: software responsabile dell'esecuzione dei container.
- **`kubelet`**: garantisce che i container siano integri e funzionanti nel loro Pod associato.
- **`kube-proxy`**: gestisce le regole della rete che consentono la comunicazione con i tuoi Pods.

Per ulteriori informazioni, consulta Nodi nella documentazione di Kubernetes.

Il cluster Amazon EKS può pianificare Pods su qualsiasi combinazione di nodi autogestiti, gruppi di nodi gestiti da Amazon EKS e AWS Fargate. Per ulteriori informazioni sui nodi implementati nel cluster, consulta Visualizzazione delle risorse Kubernetes.

> **Important**
>
> AWS Fargate con Amazon EKS non è disponibile negli AWS GovCloud Stati Uniti orientali e AWS GovCloud negli Stati Uniti occidentali.

> **Note**
>
> I nodi devono trovarsi nello stesso VPC delle sottoreti selezionate al momento della creazione del cluster, ma non necessariamente nelle stesse sottoreti.

Nella tabella seguente vengono forniti diversi criteri per valutare quali opzioni sono più adatte alle proprie esigenze. I nodi connessi creati al di fuori di Amazon EKS possono essere soltanto visualizzati e non sono inclusi in questa tabella.

> **Note**
>
> Bottlerocket presenta alcune differenze specifiche rispetto alle informazioni generali contenute in questa tabella. Per ulteriori informazioni, consulta la documentazione di Bottlerocket su GitHub.

Criteri	Gruppi di nodi gestiti EKS	Nodi autogestiti	AWS Fargate
Può essere implementato in AWS Outposts	No	Sì	No
Può essere implementato nella zona locali AWS	No	Sì: – per ulteriori informazioni, consulta Amazon EKS e zone locali AWS.	No
Può eseguire container che richiedono Windows	Sì	Sì: il cluster richiede comunque almeno un nodo Linux (due consigliati per la disponibilità).	No
Può eseguire container che richiedono Linux	Sì	Sì	Sì
Può eseguire carichi di lavoro che richiedono il chip Inferentia	Sì – Solo nodi Amazon Linux	Sì – Solo Amazon Linux	No
Può eseguire carichi di lavoro che richiedono una GPU	Sì – Solo nodi Amazon Linux	Sì – Solo Amazon Linux	No
Può eseguire carichi di lavoro che richiedono processori Arm	Sì	Sì	No
Può eseguire AWS Bottlerocket	Sì	Sì	No
I pod condividono un ambiente di runtime del kernel con altri Pods	Sì: tutti i Pods su ciascun nodo	Sì: tutti i Pods su ciascuno dei tuoi nodi	No: ogni Pod ha un kernel dedicato

Criteri	Gruppi di nodi gestiti EKS	Nodi autogestiti	AWS Fargate
I pod condividono risorse di CPU, memoria, archiviazione e rete con altri Pods.	Sì: può generare risorse inutilizzate su ciascun nodo	Sì: può generare risorse inutilizzate su ciascun nodo	No: ogni Pod dispone di risorse dedicate e può essere ridimensionato in modo indipendente per massimizzare l'utilizzo delle risorse.
I pod possono utilizzare più hardware e memoria rispetto alle specifiche dei Pod	Sì: se il Pod richiede più risorse di quelle richieste e le risorse sono disponibili sul nodo, il Pod può utilizzare risorse aggiuntive.	Sì: se il Pod richiede più risorse di quelle richieste e le risorse sono disponibili sul nodo, il Pod può utilizzare risorse aggiuntive.	No: il Pod può essere comunque reimplementato utilizzando una vCPU e una configurazione di memoria più grandi.

Criteri	Gruppi di nodi gestiti EKS	Nodi autogestiti	AWS Fargate
Deve implementare e gestire le istanze Amazon EC2	Sì: automatizzato attraverso Amazon EKS se è stata implementata un'AMI ottimizzata per Amazon EKS. Se è stata implementata un'AMI personalizzata, è necessario aggiornare l'istanza manualmente.	Sì: configurazione manuale o utilizzo dei modelli AWS CloudFormation forniti da Amazon EKS per implementare i nodi Linux (x86), Linux (Arm) o Windows.	No
È necessario proteggere, mantenere e applicare patch al sistema operativo delle istanze Amazon EC2	Sì	Sì	No
Può fornire argomenti di bootstrap all'implementazione di un nodo, come ad esempio argomenti `kubelet` aggiuntivi.	Sì: utilizzando `eksctl` o un modello di avvio con un'AMI personalizzata	Sì: per ulteriori informazioni, consulta le informazioni sull'utilizzo dello script di bootstrap su GitHub.	No

Criteri	Gruppi di nodi gestiti EKS	Nodi autogestiti	AWS Fargate
Può assegnare indirizzi IP ai Pods da un blocco CIDR diverso rispetto all'indirizzo IP assegnato al nodo.	Sì: utilizzando un modello di avvio con un'AMI personalizzata. Per ulteriori informazioni, consulta Personalizzazione di nodi gestiti con un modello di avvio.	Sì: per ulteriori informazioni, consulta Rete personalizzata per i pod.	No
Puoi eseguire SSH nel nodo	Sì	Sì	No – Non esiste un sistema operativo host del nodo su cui eseguire il SSH.
Puoi implementare un'AMI personalizzata nei nodi	Sì – Utilizzo di un modello di avvio	Sì	No
Può implementare un CNI personalizzato nei nodi	Sì – Utilizzando un modello di avvio con un'AMI personalizzata	Sì	No

Criteri	Gruppi di nodi gestiti EKS	Nodi autogestiti	AWS Fargate
É necessario aggiornare l'AMI del nodo per conto proprio	Sì – Se hai implementato un'AMI ottimizzata per Amazon EKS, riceverai una notifica nella console Amazon EKS quando gli aggiornamenti sono disponibili. È possibile eseguire l'aggiornamento con un clic nella console. Se hai implementato un'AMI personalizzata, non riceverai una notifica nella console Amazon EKS quando gli aggiornamenti sono disponibili. È necessario eseguire l'aggiornamento per conto proprio.	Sì – Utilizzo di strumenti diversi dalla console Amazon EKS. Questo perché i nodi autogestiti non possono essere gestiti con la console Amazon EKS.	No

Criteri	Gruppi di nodi gestiti EKS	Nodi autogestiti	AWS Fargate
È necessario aggiornare la versione del nodo Kubernetes per conto proprio	Sì: se hai implementato un'AMI ottimizzata per Amazon EKS, riceverai una notifica nella console Amazon EKS quando gli aggiornamenti sono disponibili. È possibile eseguire l'aggiornamento con un clic nella console. Se hai implementato un'AMI personalizzata, non riceverai una notifica nella console Amazon EKS quando gli aggiornamenti sono disponibili. È necessario eseguire l'aggiornamento per conto proprio.	Sì – Utilizzo di strumenti diversi dalla console Amazon EKS. Questo perché i nodi autogestiti non possono essere gestiti con la console Amazon EKS.	No: non sei tu a gestire i nodi.
Può utilizzare lo spazio di archiviazione Amazon EBS con i Pods	Sì	Sì	No

Criteri	Gruppi di nodi gestiti EKS	Nodi autogestiti	AWS Fargate
Può utilizzare lo spazio di archiviazione di Amazon EFS con i Pods	Sì	Sì	Sì
Può utilizzare lo spazio di archiviazione Amazon FSx per Lustre con i Pods	Sì	Sì	No
Può utilizzare Network Load Balancer per i servizi	Sì	Sì	Sì, quando si utilizza il Creazione di un Network Load Balancer
I pod possono essere eseguiti in una sottorete pubblica	Sì	Sì	No
Può assegnare diversi gruppi di sicurezza VPC a singoli Pods	Sì: solo nodi Linux	Sì: solo nodi Linux	Sì
Può eseguire Kubernetes DaemonSets	Sì	Sì	No
Supporto `HostPort` e `HostNetwork` nel manifesto Pod	Sì	Sì	No
Regione AWS disponibilità	Tutte le regioni supportate da Amazon EKS	Tutte le Regioni Amazon EKS supportate	Alcune regioni supportate da Amazon EKS
Può eseguire container su host dedicati Amazon EC2	Sì	Sì	No

Criteri	Gruppi di nodi gestiti EKS	Nodi autogestiti	AWS Fargate
Prezzi	Costo dell'istanza Amazon EC2 che esegue più Pods. Per ulteriori informazioni, consulta Prezzi di Amazon EC2.	Costo dell'istanza Amazon EC2 che esegue più Pods. Per ulteriori informazioni, consulta Prezzi di Amazon EC2.	Costo di una singola configurazione di memoria e CPU Fargate. Ogni Pod ha il proprio costo. Per ulteriori informazioni, consultare Prezzi di AWS Fargate.

Gruppi di nodi gestiti

I gruppi di nodi gestiti Amazon EKS automatizzano il provisioning e la gestione del ciclo di vita dei nodi (istanze Amazon EC2) per i cluster Kubernetes Amazon EKS.

Con i gruppi di nodi gestiti Amazon EKS, non devi eseguire separatamente il provisioning o la registrazione delle istanze Amazon EC2 che forniscono capacità di calcolo per l'esecuzione delle tue applicazioni Kubernetes. È possibile creare, aggiornare o terminare automaticamente i nodi per il cluster con una singola operazione. Gli aggiornamenti e le interruzioni dei nodi svuotano automaticamente i nodi per garantire che le applicazioni rimangano disponibili.

Ogni nodo gestito viene assegnato come parte di un gruppo Amazon EC2 Auto Scaling gestito per l'utente da Amazon EKS. Ogni risorsa, incluse le istanze e i gruppi Auto Scaling, viene eseguita all'interno dell'account AWS . Ogni gruppo di nodi viene eseguito su più zone di disponibilità definite.

Puoi aggiungere un gruppo di nodi gestiti a cluster nuovi o esistenti utilizzando la console Amazon EKS`eksctl`, AWS CLI l' AWS API o l'infrastruttura come strumenti di codice, tra cui AWS CloudFormation. I nodi avviati come parte di un gruppo di nodi gestiti vengono automaticamente taggati per l'individuazione automatica tramite il cluster Autoscaler di Kubernetes. È possibile utilizzare il gruppo di nodi per applicare le etichette Kubernetes ai nodi e aggiornarle in qualsiasi momento.

Non ci sono costi aggiuntivi per l'utilizzo dei gruppi di nodi gestiti Amazon EKS, ci sono costi solo per le risorse AWS di cui si effettua il provisioning. Questi includono istanze Amazon EC2, volumi

Amazon EBS, orari dei cluster Amazon EKS e qualsiasi altra infrastruttura. AWS Non sono previste tariffe minime né impegni anticipati.

Per iniziare a utilizzare un nuovo cluster Amazon EKS e un gruppo di nodi gestiti, vedi Guida introduttiva ad Amazon EKS AWS Management Console e AWS CLI.

Per aggiungere un gruppo di nodi gestiti a un cluster esistente, vedi Creazione di un gruppo di nodi gestiti.

Concetti sui gruppi di nodi gestiti

- I gruppi di nodi gestiti Amazon EKS creano e gestiscono le istanze Amazon EC2 per conto dell'utente.
- Ogni nodo gestito viene assegnato come parte di un gruppo Amazon EC2 Auto Scaling gestito per l'utente da Amazon EKS. Inoltre, ogni risorsa, incluse le istanze Amazon EC2 e i gruppi di Auto Scaling, viene eseguita all'interno del tuo account. AWS
- Il gruppo Auto Scaling di un gruppo di nodi gestiti si estende su ogni sottorete specificata al momento della creazione del gruppo.
- Amazon EKS applica tag alle risorse del gruppo di nodi gestiti in modo che siano configurate per utilizzare il Cluster Autoscaler di Kubernetes.

  > **Important**
  >
  > Se esegui un'applicazione stateful in più zone di disponibilità supportate dai volumi Amazon EBS e che utilizza Scalabilità automatica di Kubernetes, devi configurare più gruppi di nodi, ognuno dei quali definito per una singola zona di disponibilità. Inoltre, è necessario abilitare la funzionalità `--balance-similar-node-groups`.

- È possibile utilizzare un modello di avvio personalizzato per un maggiore livello di flessibilità e personalizzazione durante l'implementazione dei nodi gestiti. Ad esempio, è possibile specificare argomenti `kubelet` aggiuntivi e utilizzare un'AMI personalizzata. Per ulteriori informazioni, consulta Personalizzazione di nodi gestiti con un modello di avvio. Se non si utilizza un modello di avvio personalizzato quando si crea per la prima volta un gruppo di nodi gestiti, è disponibile un modello di avvio generato automaticamente. Non modificare manualmente questo modello generato automaticamente o si verificano errori.
- Amazon EKS segue il modello di responsabilità condivisa per i CVE e le patch di sicurezza sui gruppi di nodi gestiti. Quando i nodi gestiti eseguono un'AMI ottimizzata per Amazon EKS, Amazon EKS è responsabile della creazione di versioni con patch dell'AMI quando vengono segnalati bug

o problemi. A questo scopo vengono pubblicate delle correzioni. L'utente è invece responsabile della distribuzione di queste versioni AMI con patch ai gruppi di nodi gestiti. Quando i nodi gestiti eseguono un'AMI personalizzata, l'utente è responsabile della creazione di versioni con patch dell'AMI quando vengono segnalati bug o problemi, oltre che della distribuzione dell'AMI. Per ulteriori informazioni, consulta Aggiornamento di un gruppo di nodi gestiti.

- I gruppi di nodi gestiti Amazon EKS possono essere avviati in sottoreti pubbliche e private. Se si avvia un gruppo di nodi gestiti in una sottorete pubblica in data 22 aprile 2020 o successiva, sarà necessario che la sottorete abbia `MapPublicIpOnLaunch` impostato su VERO affinché le istanze possano partecipare correttamente a un cluster. Se la sottorete pubblica è stata creata utilizzando `eksctl` o i AWS CloudFormation modelli forniti da Amazon EKS a partire dal 26 marzo 2020, questa impostazione è già impostata su true. Se le sottoreti pubbliche sono state create prima del 26 marzo 2020 devi modificare manualmente l'impostazione. Per ulteriori informazioni, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv4` pubblici per la sottorete.
- Quando implementi un gruppo di nodi gestito in sottoreti private, devi assicurarti che possa accedere ad Amazon ECR per estrarre le immagini dei container. È possibile farlo collegando un gateway NAT alla tabella di routing della sottorete o aggiungendo i seguenti endpoint VPC AWS PrivateLink:
  - Interfaccia endpoint dell'API di Amazon ECR: `com.amazonaws.`*`region-code`*`.ecr.api`
  - Interfaccia endpoint dell'API del registro Docker di Amazon ECR: `com.amazonaws.`*`region-code`*`.ecr.dkr`
  - Endpoint del gateway Amazon S3: `com.amazonaws.`*`region-code`*`.s3`

  Per altri servizi ed endpoint di uso comune, consulta la sezione Requisiti dei cluster privati.
- I gruppi di nodi gestiti non possono essere distribuiti su AWS Outpostso all'interno AWS Wavelength delle nostre AWS Local Zones.
- È possibile creare più gruppi di nodi gestiti all'interno di un singolo cluster. Ad esempio, puoi creare un gruppo di nodi con l'AMI Amazon Linux standard ottimizzata per Amazon EKS per alcuni carichi di lavoro e un altro con la variante GPU per carichi di lavoro che richiedono il supporto GPU.
- Se il gruppo di nodi gestiti rileva un errore di controllo dello stato dell'istanza Amazon EC2, Amazon EKS restituirà un messaggio di errore che consente di diagnosticare il problema. Per ulteriori informazioni, consulta Codici di errore dei gruppi di nodi gestiti.
- Amazon EKS aggiunge le etichette Kubernetes alle istanze del gruppo di nodi gestiti. Queste etichette fornite da Amazon EKS hanno il prefisso `eks.amazonaws.com`.
- Amazon EKS svuota automaticamente i nodi utilizzando l'API Kubernetes durante le interruzioni o gli aggiornamenti.

- I budget di interruzione del pod non vengono rispettati quando si termina un nodo con `AZRebalance` o si riduce il numero di nodi desiderato. Queste operazioni cercano di espellere Pods dal nodo. Tuttavia, se trascorrono più di 15 minuti, il nodo viene terminato a prescindere dal fatto che tutti i Pods sul nodo siano stati terminati. Per prolungare il periodo fino alla terminazione del nodo, aggiungi un hook del ciclo di vita al gruppo con scalabilità automatica. Per ulteriori informazioni, consulta Aggiungere un hook del ciclo di vita nella Guida per l'utente di Dimensionamento automatico Amazon EC2.
- Per eseguire correttamente il processo di scarico dopo aver ricevuto una notifica di interruzione spot o una notifica di ribilanciamento della capacità, `CapacityRebalance` deve essere impostato su `true`.
- L'aggiornamento di gruppi di nodi gestiti rispettano i budget di interruzione Pod impostati per i Pods. Per ulteriori informazioni, consulta Comportamento dell'aggiornamento del nodo gestito.
- I gruppi di nodi gestiti Amazon EKS non prevedono costi aggiuntivi. Paghi solo per le AWS risorse che fornisci.
- Se si desidera crittografare i volumi Amazon EBS per i nodi, è possibile implementare i nodi utilizzando un modello di avvio. Per implementare nodi gestiti con volumi Amazon EBS crittografati senza utilizzare un modello di avvio, crittografare tutti i nuovi volumi Amazon EBS creati nell'account. Per ulteriori informazioni, consulta Encryption by default nella Amazon EC2 User Guide.

Tipi di capacità del gruppo di nodi gestiti

Quando si crea un gruppo di nodi gestito, è possibile scegliere il tipo di capacità su on demand o Spot. Amazon EKS implementa un gruppo di nodi gestito con un gruppo di Dimensionamento automatico Amazon EC2 che contiene solo istanze on demand o solo istanze spot Amazon EC2. All'interno di un singolo cluster Kubernetes, puoi pianificare i Pods per le applicazioni a tolleranza d'errore su gruppi di nodi gestiti Spot e per le applicazioni non a tolleranza d'errore su gruppi di nodi on demand. Per impostazione predefinita, un gruppo di nodi gestito distribuisce istanze Amazon EC2 on demand.

On demand

Con Istanze on demand, sono previsti costi per la capacità di calcolo entro la seconda ora senza impegni a lungo termine.

Come funziona

Per impostazione predefinita, se non specifichi un Capacity Type (Tipo di capacità), il provisioning del gruppo di nodi gestiti viene eseguito con istanze on demand. Un gruppo di nodi gestiti configura un gruppo Amazon EC2 Auto Scaling per conto dell'utente con le seguenti impostazioni applicate:

- La strategia di allocazione per il provisioning della capacità On-Demand è impostata su `prioritized`. Il gruppo di nodi gestiti utilizza l'ordine dei tipi di istanze approvata dall'API per determinare quale tipo di istanza utilizzare prima per soddisfare la capacità on demand. Ad esempio, è possibile specificare tre tipi di istanza nell'ordine seguente: `c5.large`, `c4.large`, e `c3.large`. Quando le istanze on demand vengono avviate, il gruppo di nodi gestiti soddisfa la capacità on demand a partire da `c5.large`, quindi `c4.large`, e infine `c3.large`. Per maggiori informazioni, consultare Gruppo Amazon EC2 Auto Scaling nella Guida per l'utente di Amazon EC2 Auto Scaling.
- Amazon EKS aggiunge l'etichetta Kubernetes seguente a tutti i nodi del gruppo di nodi gestiti che specifica il tipo di capacità: `eks.amazonaws.com/capacityType: ON_DEMAND`. È possibile utilizzare questa etichetta sui nodi on demand per pianificare applicazioni con stato o fault intolerant.

Spot

Le istanze Spot di Amazon EC2 sono capacità Amazon EC2 di riserva che offrono sconti elevati rispetto ai prezzi on demand. Quando EC2 necessita nuovamente della capacità in uso nelle istanze Spot Amazon EC2, questa essere interrotta con una notifica di interruzione di due minuti. Per ulteriori informazioni, consulta le istanze Spot nella Guida per l'utente di Amazon EC2. É possibile configurare un gruppo di nodi gestito con Istanze spot Amazon EC2 per ottimizzare i costi per i nodi di calcolo in esecuzione nel tuo cluster Amazon EKS.

Come funziona

Per utilizzare istanze Spot all'interno di un gruppo di nodi gestiti, creare un gruppo di nodi gestiti impostando il tipo di capacità come `spot`. Applicando le seguenti best practices Spot, un gruppo di nodi gestiti configura un gruppo Amazon EC2 Auto Scaling per conto dell'utente:

- Per garantire che il provisioning dei nodi Spot venga eseguito nei pool di capacità Spot ottimali, la strategia di allocazione è impostata su uno dei seguenti valori:
  - `price-capacity-optimized` (PCO): quando si creano nuovi gruppi di nodi in un cluster con Kubernetes versione `1.28` o successive, la strategia di allocazione è impostata su `price-`

`capacity-optimized`. Tuttavia, la strategia di allocazione non viene modificata per i gruppi di nodi già creati con `capacity-optimized` prima che i gruppi di nodi gestiti da Amazon EKS inizino a supportare PCO.

- `capacity-optimized` (CO): quando si creano nuovi gruppi di nodi in un cluster con Kubernetes versione `1.27` o successive, la strategia di allocazione è impostata su `capacity-optimized`.

Per aumentare il numero di pool di capacità Spot disponibili da cui allocare le capacità, configurare un gruppo di nodi gestiti per l'utilizzo di più tipi di istanza.

- Amazon EC2 Spot Capacity Rebalancing è abilitato in modo che Amazon EKS possa svuotare e bilanciare correttamente i nodi Spot per ridurre al minimo le interruzioni delle applicazioni quando un nodo Spot è a rischio elevato di interruzione. Per maggiori informazioni, consultare Rebalancing Capacity di Amazon EC2 Auto Scaling nella Guida per l'utente di Amazon EC2 Auto Scaling.
  - Quando un nodo Spot riceve un consiglio di ribilanciamento, Amazon EKS tenta automaticamente di avviare un nuovo nodo Spot sostitutivo.
  - Se un avviso di interruzione Spot di due minuti arriva prima che il nodo Spot sostitutivo si trovi in uno stato `Ready`, Amazon EKS inizia a svuotare il nodo Spot che ha ricevuto il suggerimento di ribilanciamento. Amazon EKS svuota il nodo nel modo migliore possibile. Di conseguenza, non è assicurato che Amazon EKS attenda che il nodo sostitutivo si unisca al cluster prima di svuotare il nodo esistente.
  - Quando un nodo Spot sostitutivo viene avviato ed è allo stato `Ready` su Kubernetes, Amazon EKS isola e svuota il nodo Spot che ha ricevuto il suggerimento di ribilanciamento. L'isolamento del nodo Spot assicura che il controller di servizio non invii nuove richieste a questo nodo Spot. Il nodo viene rimosso anche dall'elenco elenco di nodi Spot sani e attivi. Lo svuotamento del nodo Spot assicura che i Pods in esecuzione vengano espulsi in modo aggraziato.
- Amazon EKS aggiunge l'etichetta Kubernetes seguente a tutti i nodi del gruppo di nodi gestiti che specifica il tipo di capacità: `eks.amazonaws.com/capacityType: SPOT`. È possibile utilizzare questa etichetta per pianificare applicazioni fault tolerant sui nodi Spot.

Considerazioni per la selezione di un tipo di capacità

Quando si decide se implementare un gruppo di nodi con capacità On-Demand o Spot, è necessario considerare le seguenti condizioni:

- Le istanze Spot sono consigliate per applicazioni senza stato, fault tolerant e flessibili. Queste includono carichi di lavoro di formazione batch e machine learning, ETL di big data come Apache

Spark, applicazioni di elaborazione delle code ed endpoint API senza stato. Poiché Spot è una capacità Amazon EC2 di riserva che può cambiare nel tempo, consigliamo di utilizzare la capacità Spot per carichi di lavoro con tolleranza di interruzione. Più specificamente, la capacità Spot è adatta per carichi di lavoro in grado di tollerare periodi in cui la capacità richiesta non è disponibile.

- Consigliamo di utilizzare On-demand per le applicazioni che sono fault intolerant. Ciò include strumenti di gestione dei cluster come strumenti di monitoraggio e operativi, implementazioni che richiedono `StatefulSets` e applicazioni con stato, come database.
- Per ottimizzare la disponibilità delle applicazioni durante l'utilizzo di istanze Spot, è consigliabile configurare un gruppo di nodi gestiti Spot per l'utilizzo di più tipi di istanza. Quando si utilizzano più tipi di istanza, si consiglia di applicare le seguenti regole:
  - All'interno di un gruppo di nodi gestiti, se si utilizza il Cluster Autoscaler, si consiglia di utilizzare un set flessibile di tipi di istanza con la stessa quantità di vCPU e risorse di memoria. Questo per garantire che i nodi del cluster vengano dimensionati come previsto. Ad esempio, se sono necessarie quattro vCPU e otto memorie GiB, utilizzare `c3.xlarge`, `c4.xlarge`, `c5.xlarge`, `c5d.xlarge`, `c5a.xlarge`, `c5n.xlarge` o altri tipi di istanza simili.
  - Per migliorare la disponibilità delle applicazioni, si consiglia di implementare più gruppi di nodi gestiti Spot. Per questo, ogni gruppo dovrebbe utilizzare un set flessibile di tipi di istanza con le stesse risorse vCPU e memoria. Ad esempio, se sono necessarie 4 vCPU e 8 memorie GiB, si consiglia di creare un gruppo di nodi gestiti con `c3.xlarge`, `c4.xlarge`, `c5.xlarge`, `c5d.xlarge`, `c5a.xlarge`, `c5n.xlarge` o altri tipi di istanza simili; e un secondo gruppo di nodi gestiti con `m3.xlarge`, `m4.xlarge`, `m5.xlarge`, `m5d.xlarge`, `m5a.xlarge`, `m5n.xlarge` o altri tipi di istanza simili.
  - Quando si implementa il gruppo di nodi con il tipo di capacità Spot che utilizza un modello di avvio personalizzato, utilizzare l'API per passare più tipi di istanza. Non passare un singolo tipo di istanza tramite il modello di avvio. Per ulteriori informazioni sull'implementazione di un gruppo di nodi tramite un modello di avvio, vedere Personalizzazione di nodi gestiti con un modello di avvio.

Creazione di un gruppo di nodi gestiti

In questo argomento viene descritto come avviare gruppi di nodi gestiti Amazon EKS per nodi che si registrano con il cluster Amazon EKS. Dopo che i nodi vengono aggiunti al cluster, puoi implementare le applicazioni Kubernetes per gli stessi.

Se è la prima volta che avvii un gruppo di nodi gestiti Amazon EKS, consigliamo invece di seguire una delle nostre guide Guida introduttiva ad Amazon EKS. Le guide forniscono procedure dettagliate per la creazione di un cluster Amazon EKS con nodi.

Important

- I nodi Amazon EKS sono istanze standard Amazon EC2 standard. La fatturazione viene effettuata in base ai consueti prezzi Amazon EC2. Per ulteriori informazioni, consulta Prezzi di Amazon EC2.
- Non puoi creare nodi gestiti in un ambiente in Regione AWS cui hai AWS Outposts o abilitato AWS Local Zones. AWS WavelengthPuoi creare nodi autogestiti in un ambiente in Regione AWS cui hai AWS Outposts abilitato AWS Wavelength hai abilitato AWS Local Zones. Per ulteriori informazioni, consulta Avvio di nodi Amazon Linux autogestiti, Avvio dei nodi Windows autogestiti e Avvio dei nodi Bottlerocket autogestiti. Puoi anche creare un gruppo di nodi Amazon Linux autogestito su un Outpost. Per ulteriori informazioni, consulta Avvio di nodi Amazon Linux autogestiti su un Outpost.
- Se non specifichi un ID AMI per il file `bootstrap.sh` incluso in Linux o Bottlerocket ottimizzato per Amazon EKS, i gruppi di nodi gestiti impongono un numero massimo sul valore di `maxPods`. Per le istanze con meno di 30 vCPU il numero massimo è `110`. Per le istanze con più di 30 vCPU il numero massimo passa a `250`. Questi numeri si basano sulle soglie di scalabilità di Kubernetes e sulle impostazioni consigliate dai test interni del team di scalabilità di Amazon EKS. Per ulteriori informazioni, consulta il post del blog dedicato all'aumento dei limiti dei pod per nodo del plug-in CNI di Amazon VPC.

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Creazione di un cluster Amazon EKS.
- Un ruolo IAM esistente per i nodi da utilizzare. Per crearne uno, consulta Ruolo IAM del nodo Amazon EKS. Se questo ruolo non prevede nessuna delle policy per VPC CNI, per i pod VPC CNI è necessario il ruolo separato riportato di seguito.
- (Facoltativo, ma consigliato) Il componente aggiuntivo Amazon VPC CNI plugin for Kubernetes configurato con il suo personale ruolo IAM a cui è allegata la policy IAM necessaria. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

- Familiarità con le considerazioni riportate in Scelta di un tipo di istanza Amazon EC2. A seconda del tipo di istanza scelto, potrebbero esserci ulteriori prerequisiti per il cluster e il VPC.
- Per aggiungere un gruppo di nodi gestiti da Windows, devi prima abilitare il supporto Windows per il tuo cluster. Per ulteriori informazioni, consulta Abilitazione del supporto di Windows per il cluster Amazon EKS.

È possibile creare un gruppo di nodi gestiti con `eksctl` o il AWS Management Console.

eksctl

Creazione di un gruppo di nodi gestito con **`eksctl`**

Questa procedura richiede `eksctl` versione `0.183.0` o versioni successive. Puoi verificare la versione con il comando seguente:

```
eksctl version
```

Per istruzioni sull'installazione o sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

1. (Facoltativo) Se la policy IAM gestita AmazonEKS_CNI_Policy è collegata al Ruolo IAM del nodo Amazon EKS, consigliamo, invece, di assegnarla a un ruolo IAM associato all'account di servizio Kubernetes `aws-node`. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).
2. Creare un gruppo di nodi gestiti con o senza utilizzare un modello di avvio personalizzato. La specifica manuale di un modello di avvio consente una maggiore personalizzazione di un gruppo di nodi. Ad esempio, può consentire l'implementazione di un'AMI personalizzata o la presenza di argomenti sullo script `boostrap.sh` in un'AMI ottimizzata per Amazon EKS. Per avere un elenco completo di tutte le opzioni e le impostazioni predefinite disponibili, inserisci il comando seguente.

   ```
   eksctl create nodegroup --help
   ```

   Nel comando seguente, sostituisci *`my-cluster`* con il nome del tuo cluster e sostituisci *`my-mng`* con il nome del gruppo di nodi. Il nome del gruppo di nodi non può contenere più

di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura.

> **Important**
>
> Se non utilizzi un modello di avvio personalizzato durante la prima creazione di un gruppo di nodi gestiti, non utilizzarne uno per il gruppo di nodi in un secondo momento. Se non hai specificato un modello di avvio personalizzato, il sistema genera automaticamente un modello di avvio che non è consigliabile modificare manualmente. La modifica manuale di questo modello di avvio generato in automatico potrebbe causare errori.

Senza uh modello di avvio

`eksctl` crea un modello di avvio Amazon EC2 predefinito nell'account e implementa il gruppo di nodi utilizzando un modello di avvio creato in base alle opzioni specificate. Prima di specificare un valore per `--node-type`, consulta Scelta di un tipo di istanza Amazon EC2.

Sostituisci *`ami-family`* con una parola chiave consentita. Per ulteriori informazioni, consulta Setting the node AMI Family (Impostazione della famiglia AMI dei nodi) nella documentazione di `eksctl`. Sostituisci *`my-key`* con il nome della coppia di chiavi Amazon EC2 o della chiave pubblica. Questa chiave viene utilizzata per eseguire il SSH nei nodi dopo il loro avvio.

> **Note**
>
> Per Windows, questo comando non abilita SSH. Associa, invece, la coppia di chiavi Amazon EC2 all'istanza e ti consente di eseguire il protocollo RDP nell'istanza.

Se non disponi di una coppia di chiavi Amazon EC2, puoi crearla nella AWS Management Console. Per Linux informazioni, consulta le coppie di chiavi e le Linux istanze di Amazon EC2 nella Guida per l'utente di Amazon EC2. Per Windows informazioni, consulta le coppie di chiavi e le Windows istanze di Amazon EC2 nella Guida per l'utente di Amazon EC2.

Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

- Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
- No, Pods nel cluster è richiesto l'accesso al servizio di metadati dell'istanza Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.

Se desideri bloccare l'accesso del Pod a IMDS, aggiungi l'opzione **`--disable-pod-imds`** al seguente comando.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --region region-code \
  --name my-mng \
  --node-ami-family ami-family \
  --node-type m5.large \
  --nodes 3 \
  --nodes-min 2 \
  --nodes-max 4 \
  --ssh-access \
  --ssh-public-key my-key
```

Le istanze possono assegnare facoltativamente un numero significativamente superiore di indirizzi IP ai Pods, assegna gli indirizzi IP ai Pods da un blocco CIDR diverso da quello dell'istanza ed essere distribuite in un cluster senza accesso a Internet. Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2, Rete personalizzata per i pod, e Requisiti dei cluster privati per ulteriori opzioni da aggiungere al comando precedente.

In base al tipo di istanza, i gruppi di nodi gestiti calcolano e applicano un singolo valore per il numero massimo di Pods che possono essere eseguiti su ogni nodo del gruppo. Se crei un gruppo di nodi con tipi di istanza diversi, il valore più piccolo calcolato in tutti i tipi di istanza viene applicato come numero massimo di Pods che possono essere eseguiti su ogni tipo di istanza nel gruppo di nodi. I gruppi di nodi gestiti calcolano il valore utilizzando lo script a cui si fa riferimento in Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods.

Con un modello di avvio

Il modello di avvio deve già esistere e deve soddisfare i requisiti specificati in Informazioni di base sulla configurazione del modello di avvio.

Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

- Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
- No, Pods nel cluster è richiesto l'accesso al servizio di metadati dell'istanza Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.

Se desideri bloccare l'accesso del Pod a IMDS, specifica le impostazioni necessarie nel modello di avvio.

a. Copia i seguenti contenuti sul dispositivo. Sostituisci i *example values*, quindi esegui il comando modificato per creare il file `eks-nodegroup.yaml`. Diverse impostazioni specificate durante l'implementazione senza un modello di avvio vengono spostate nel modello di avvio. Se non si specifica un `version`, viene utilizzata la versione di default del modello.

```
cat >eks-nodegroup.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
managedNodeGroups:
- name: my-mng
  launchTemplate:
    id: lt-id
    version: "1"
EOF
```

Per un elenco completo delle impostazioni del file di configurazione `eksctl`, consulta Schema dei file config nella documentazione di `eksctl`. Facoltativamente, le istanze possono assegnare un numero significativamente più elevato di indirizzi IP ai Pods, assegnare indirizzi IP ai Pods da un blocco CIDR diverso da quello dell'istanza,

utilizzare il runtime `containerd` ed essere implementate in un cluster senza accesso a Internet in uscita. Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2, Rete personalizzata per i pod, Prova la migrazione da Docker a `containerd`, e Requisiti dei cluster privati per ulteriori opzioni da aggiungere al file di configurazione.

Se non è stato specificato un ID AMI nel modello di avvio, i gruppi di nodi gestiti calcolano e applicano un singolo valore per il numero massimo di Pods che possono essere eseguiti su ciascun nodo del gruppo di nodi, in base al tipo di istanza. Se crei un gruppo di nodi con tipi di istanza diversi, il valore più piccolo calcolato in tutti i tipi di istanza viene applicato come numero massimo di Pods che possono essere eseguiti su ogni tipo di istanza nel gruppo di nodi. I gruppi di nodi gestiti calcolano il valore utilizzando lo script a cui si fa riferimento in Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods.

Se nel modello di avvio è stato specificato un ID AMI, specifica il numero massimo di Pods che possono essere eseguiti su ciascun nodo del gruppo se utilizzi reti personalizzate o se desideri aumentare il numero di indirizzi IP assegnati all'istanza. Per ulteriori informazioni, consulta Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods.

b. Implementare il gruppo di nodi mediante il comando seguente.

```
eksctl create nodegroup --config-file eks-nodegroup.yaml
```

AWS Management Console

Per creare un gruppo di nodi gestito utilizzando AWS Management Console

1. Attendi che lo stato del cluster risulti `ACTIVE`. Non è possibile creare un gruppo di nodi gestiti per un cluster non ancora `ACTIVE`.
2. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
3. Scegli il nome del cluster in cui desideri creare un gruppo di nodi gestiti.
4. Seleziona la scheda Compute (Calcolo).
5. Scegli Add node group (Aggiungi gruppo di nodi).

6. Nella pagina Configura il gruppo di nodi, compila i parametri di conseguenza e quindi scegli Successivo.

   - Nome: inserisci un nome univoco per il gruppo di nodi gestiti. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura.
   - Ruolo IAM del nodo: scegli il ruolo dell'istanza del nodo da utilizzare con il gruppo di nodi. Per ulteriori informazioni, consulta Ruolo IAM del nodo Amazon EKS.

     > **Important**
     > - Non è possibile utilizzare lo stesso ruolo utilizzato per creare i cluster.
     > - Consigliamo di utilizzare un ruolo che non è attualmente in uso da alcun gruppo di nodi autogestiti. In caso contrario, prevedi l'utilizzo di un nuovo gruppo di nodi autogestiti. Per ulteriori informazioni, consulta Eliminazione di un gruppo di nodi gestiti.

   - Usa modello di avvio (Facoltativo): scegli se desideri utilizzare un modello di avvio esistente. Seleziona un Launch Template Name (Nome del modello di avvio). Quindi, seleziona una Launch template version (Versione del modello di avvio). Se non selezioni una versione, Amazon EKS utilizza la versione predefinita del modello. I modelli di avvio permettono una maggiore personalizzazione del gruppo di nodi, ad esempio consentono di implementare un'AMI personalizzata, assegnare un numero significativamente più elevato di indirizzi IP ai Pods, assegnare indirizzi IP ai Pods da un blocco CIDR diverso da quello dell'istanza, abilitare il runtime `containerd` per le istanze e implementando i nodi in un cluster senza accesso a Internet in uscita. Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2, Rete personalizzata per i pod, Prova la migrazione da Docker a `containerd` e Requisiti dei cluster privati.

     Il modello di avvio deve soddisfare i requisiti in Personalizzazione di nodi gestiti con un modello di avvio. Se non usi il tuo modello di avvio, l'API di Amazon EKS crea un modello di avvio Amazon EC2 predefinito nel tuo account e implementa il gruppo di nodi utilizzando il modello di avvio predefinito.

     Se implementi i ruoli IAM per gli account di servizio, assegna le autorizzazioni necessarie direttamente a tutti i Pod che richiedono l'accesso ai servizi AWS . Se nessun Pods nel cluster richiede l'accesso a IMDS per altri motivi, ad esempio il recupero della Regione AWS corrente, puoi disabilitare l'accesso a IMDS per i Pods che non utilizzano la rete host

in un modello di avvio. Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.

- Etichette Kubernetes (Facoltativo): puoi scegliere di applicare etichette Kubernetes ai nodi del gruppo di nodi gestiti.
- Taint Kubernetes (Facoltativo): puoi scegliere di applicare i taint Kubernetes ai nodi del gruppo di nodi gestiti. Nel menu Effetto sono disponibili le opzioni **`NoSchedule`**, **`NoExecute`** e **`PreferNoSchedule`**. Per ulteriori informazioni, consulta Taint dei nodi nei gruppi di nodi gestiti.
- Tag: (facoltativo) è possibile scegliere di aggiungere tag al gruppo di nodi gestiti Amazon EKS. Questi tag non si propagano ad altre risorse del gruppo di nodi, ad esempio gruppi Auto Scaling o istanze. Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Amazon EKS.

7. Nella pagina Imposta configurazione di calcolo e dimensionamento, compila i parametri di conseguenza e quindi scegli Successivo.

   - Tipo di AMI: seleziona un tipo di AMI. Se stai implementando istanze Arm, accertati di valutare le considerazioni in AMI Amazon Linux Arm ottimizzate per Amazon EKS prima dell'implementazione.

     Se nella pagina precedente hai specificato un modello di avvio e un'AMI nel modello di avvio, non puoi selezionare un valore. Viene visualizzato il valore del modello. L'AMI specificata nel modello deve soddisfare i requisiti indicati in Specifica di un'AMI.
   - Tipo di capacità: seleziona un tipo di capacità. Per ulteriori informazioni sulla scelta di un tipo di capacità, consulta Tipi di capacità del gruppo di nodi gestiti. Non è possibile combinare diversi tipi di capacità all'interno dello stesso gruppo di nodi. Se desideri utilizzare entrambi i tipi di capacità, crea gruppi di nodi separati, ognuno con i propri tipi di capacità e istanza.
   - Tipi di istanza: per impostazione predefinita, viene specificato uno o più tipi di istanza. Per rimuovere un tipo di istanza predefinito, seleziona il `X` sul lato destro del tipo di istanza. Scegli il tipo di istanza da utilizzare nel gruppo di nodi gestiti. Per ulteriori informazioni, consulta Scelta di un tipo di istanza Amazon EC2.

     Nella console viene visualizzato un insieme di tipi di istanza di uso comune. Se devi creare un gruppo di nodi gestiti con un tipo di istanza non visualizzato, utilizza `eksctl`, la AWS CLI, AWS CloudFormation o un SDK per creare il gruppo di nodi. Se nella pagina precedente è stato specificato un modello di avvio, non è possibile selezionare un valore

perché il tipo di istanza deve essere specificato nel modello di avvio. Viene visualizzato il valore del modello di avvio. Se è stato selezionato Spot (Spot) per Capacity type (Tipo di capacità), al fine di migliorare la disponibilità è consigliabile specificare più tipi di istanza.

- Dimensioni disco: inserisci le dimensioni del disco (in GiB) da utilizzare per il volume root del nodo.

  Se nella pagina precedente hai specificato un modello di avvio, non puoi selezionare un valore perché deve essere specificato nel modello di avvio.

- Dimensione desiderata: specifica il numero corrente di nodi che il gruppo di nodi gestiti deve mantenere all'avvio.

  > **Note**
  >
  > Amazon EKS non dimensiona in automatico il gruppo di nodi. Tuttavia, è possibile configurare il Cluster Autoscaler di Kubernetes per eseguire questa operazione.

- Dimensione minima: specifica il numero minimo di nodi a cui il gruppo di nodi gestiti può essere ridotto.
- Dimensione massima: specifica il numero massimo di nodi a cui il gruppo di nodi gestiti può essere aumentato.
- Configurazione dell'aggiornamento dei gruppi di nodi: (facoltativo) puoi selezionare il numero o la percentuale di nodi da aggiornare in parallelo. Questi nodi non saranno disponibili durante l'aggiornamento. Per Numero massimo non disponibile, seleziona una delle seguenti opzioni e specifica un Valore:
  - Numero: seleziona e specifica il numero di nodi nel gruppo nodi che possono essere aggiornati in parallelo.
  - Percentuale: seleziona e specifica la percentuale di nodi nel gruppo di nodi che possono essere aggiornati in parallelo. Questa funzione è utile se disponi di un numero elevato di nodi nel gruppo di nodi.

8. Nella pagina Specifica reti, compila i parametri opportunamente, quindi scegli Successivo.

   - Sottoreti: sceglii le sottoreti in cui avviare i nodi gestiti.

     > **Important**
     >
     > Se esegui un'applicazione stateful in più zone di disponibilità supportate dai volumi Amazon EBS e che utilizza Scalabilità automatica di Kubernetes, devi configurare

più gruppi di nodi, ognuno dei quali definito per una singola zona di disponibilità. Inoltre, è necessario abilitare la funzionalità `--balance-similar-node-groups`.

> **Important**
>
> - Se si sceglie una sottorete pubblica e nel cluster è abilitato solo l'endpoint del server API pubblico, è necessario che la sottorete disponga di `MapPublicIPOnLaunch` impostato su `true` per consentire alle istanze di unirsi correttamente a un cluster. Se la sottorete è stata creata utilizzando `eksctl` o i modelli AWS CloudFormation forniti da Amazon EKS in data 26 marzo 2020 successiva, questa impostazione è già impostata su `true`. Se le sottoreti sono state create con `eksctl` o i AWS CloudFormation modelli prima del 26 marzo 2020, è necessario modificare l'impostazione manualmente. Per ulteriori informazioni, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv4` pubblici per la sottorete.
> - Se utilizzi un modello di avvio e specifichi più interfacce di rete, Amazon EC2 non assegnerà automaticamente un indirizzo `IPv4` pubblico, anche se `MapPublicIpOnLaunch` è impostato su `true`. Affinché i nodi possano unirsi al cluster in questo scenario, è necessario abilitare l'endpoint del server API privato del cluster, oppure avviare i nodi in una sottorete privata con accesso Internet in uscita fornito attraverso un metodo alternativo, ad esempio un gateway NAT. Per ulteriori informazioni, consulta la sezione Indirizzamento IP delle istanze Amazon EC2 nella Guida per l'utente di Amazon EC2.

- Configurazione dell'accesso SSH ai nodi (facoltativo). L'abilitazione di SSH consente di connettersi alle istanze e raccogliere informazioni diagnostiche in caso di problemi. Consigliamo vivamente di abilitare l'accesso remoto quando crei un gruppo di nodi. Non è possibile abilitare l'accesso remoto dopo la creazione del gruppo di nodi.

  Se scegli di utilizzare un modello di avvio, questa opzione non viene visualizzata. Per abilitare l'accesso remoto ai nodi, specifica una coppia di chiavi nel modello di avvio e assicurati che la porta appropriata sia aperta ai nodi nei gruppi di sicurezza specificati nel modello di avvio. Per ulteriori informazioni, consulta Utilizzo di gruppi di sicurezza personalizzati.

> **Note**
>
> Per Windows, questo comando non abilita SSH. Associa, invece, la coppia di chiavi Amazon EC2 all'istanza e ti consente di eseguire il protocollo RDP nell'istanza.

- Per Coppia di chiavi SSH (facoltativo), scegli una chiave SSH Amazon EC2 da utilizzare. Per Linux informazioni, consulta le coppie di chiavi e le Linux istanze di Amazon EC2 nella Guida per l'utente di Amazon EC2. Per Windows informazioni, consulta le coppie di chiavi e le Windows istanze di Amazon EC2 nella Guida per l'utente di Amazon EC2. Se scegli di utilizzare un modello di avvio, non puoi selezionarne una. Quando viene fornita una chiave SSH di Amazon EC2 per i gruppi di nodi che utilizzano AMI Bottlerocket, viene attivato anche il container amministratore. Per ulteriori informazioni, consulta Container amministratore su GitHub.
- Per Autorizzazione di accesso remoto SSH da, se desideri limitare l'accesso a istanze specifiche, seleziona i gruppi di sicurezza associati a tali istanze. Se non si selezionano gruppi di sicurezza specifici, l'accesso SSH è consentito da qualsiasi indirizzo di Internet (`0.0.0.0/0`).

9. Nella pagina Rivedi e crea, controlla la configurazione del gruppo di nodi gestiti e scegli Crea.

   Se i nodi di lavoro non riescono a unirsi al cluster, consulta Impossibile aggiungere i nodi al cluster nella Guida alla risoluzione dei problemi.

10. Guarda lo stato dei nodi e attendi che raggiungano lo stato `Ready`.

    ```
    kubectl get nodes --watch
    ```

11. (Solo nodi GPU) Se hai scelto un tipo di istanza GPU e l'AMI accelerata ottimizzata per Amazon EKS, devi applicare il plug-in del dispositivo NVIDIA per Kubernetes come DaemonSet sul cluster con il comando seguente. Sostituisci *vX.X.X*con la versione desiderata del plugin per dispositivi NVIDIA/K8S prima di eseguire il seguente comando.

    ```
    kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/nvidia-device-plugin.yml
    ```

Ora che disponi di un cluster Amazon EKS funzionante con nodi, puoi avviare l'installazione dei componenti aggiuntivi di Kubernetes e l'implementazione di applicazioni sul cluster. I seguenti argomenti della documentazione consentono di estendere la funzionalità del cluster.

- Il principale IAM in grado di creare il cluster è l'unico principale che può effettuare chiamate al server API Kubernetes tramite `kubectl` o la AWS Management Console. Se si desidera che altri principali IAM abbiano accesso al cluster, è necessario aggiungerli. Per ulteriori informazioni, consulta Concedi l'accesso alle Kubernetes API e Autorizzazioni richieste.
- Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:
  - Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
  - No, Pods nel cluster è richiesto l'accesso al servizio di metadati dell'istanza Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

  Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.
- Scalabilità automatica: configurare il Cluster Autoscaler di Kubernetes per regolare automaticamente il numero di nodi nei gruppi di nodi.
- Implementa un'applicazione di esempio sul cluster.
- Gestione dei cluster: informazioni su come utilizzare strumenti importanti per la gestione del cluster.

Aggiornamento di un gruppo di nodi gestiti

Quando si avvia un aggiornamento di un gruppo di nodi gestiti, Amazon EKS aggiorna automaticamente i nodi, completando i passaggi elencati in Comportamento dell'aggiornamento del nodo gestito. Se si utilizza un'AMI ottimizzata per Amazon EKS, Amazon EKS applica automaticamente le patch di sicurezza più recenti e gli aggiornamenti del sistema operativo ai nodi come parte dell'ultima versione di AMI.

Esistono diversi scenari in cui è utile aggiornare la versione o la configurazione del gruppo di nodi gestiti Amazon EKS:

- Hai aggiornato la versione di Kubernetes per il cluster Amazon EKS e vuoi aggiornare i nodi per utilizzare la stessa versione di Kubernetes.
- Una nuova versione dell'AMI è disponibile per il gruppo di nodi gestiti. Per ulteriori informazioni sulle versioni AMI, consultare le seguenti sezioni:
  - AMI Amazon Linux ottimizzata per Amazon EKS
  - AMI Bottlerocket ottimizzate per Amazon EKS

  - Versioni delle AMI Windows ottimizzate per Amazon EKS
- Per regolare il conteggio minimo, massimo o desiderato delle istanze nel gruppo di nodi gestiti.
- Per aggiungere o rimuovere le etichette Kubernetes dalle istanze del gruppo di nodi gestiti.
- Vuoi aggiungere o rimuovere AWS tag dal tuo gruppo di nodi gestito.
- È necessario implementare una nuova versione di un modello di avvio con modifiche alla configurazione, ad esempio un'AMI personalizzata aggiornata.
- Hai distribuito una versione `1.9.0` o successiva del componente aggiuntivo Amazon VPC CNI, abilitato il componente aggiuntivo per la delega dei prefissi e desideri che AWS Nitro System nuove istanze in un gruppo di nodi supportino un numero significativamente maggiore di. Pods Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2.
- Hai abilitato la delega del prefisso IP per i nodi Windows e desideri che nuove istanze di AWS Nitro System in un gruppo di nodi supportino un numero significativamente maggiore di. Pods Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2.

Se esiste una versione dell'AMI più recente per la versione di Kubernetes del gruppo di nodi gestiti rispetto a quella in esecuzione nel gruppo di nodi, è possibile aggiornare quest'ultima per utilizzare la nuova versione dell'AMI. Analogamente, se nel cluster è in esecuzione una versione di Kubernetes più recente rispetto al gruppo di nodi, è possibile aggiornare il gruppo di nodi per utilizzare la versione più recente dell'AMI che corrisponde alla versione di Kubernetes del cluster.

Quando un nodo in un gruppo di nodi gestiti viene terminato a causa di un'operazione di dimensionamento o di un aggiornamento, i Pods nel nodo vengono svuotati per primi. Per ulteriori informazioni, consulta Comportamento dell'aggiornamento del nodo gestito.

Aggiornare la versione di un gruppo di nodi

È possibile aggiornare la versione di un gruppo di nodi con `eksctl` o il AWS Management Console. La versione a cui si aggiorna non può essere successiva alla versione del piano di controllo.

eksctl

Per aggiornare la versione di un gruppo di nodi con **`eksctl`**

- Aggiorna un gruppo di nodi gestiti alla stessa versione AMI più recente di Kubernetes attualmente implementata nei nodi con il comando seguente. Sostituisci ogni *`example value`* con i valori in tuo possesso.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code
```

> **Note**
>
> Se stai aggiornando un gruppo di nodi implementato con un modello di avvio a una nuova versione del modello di avvio, aggiungi `--launch-template-version` *`version-number`* al precedente comando. Il modello di avvio deve soddisfare i requisiti descritti in Personalizzazione di nodi gestiti con un modello di avvio. Se il modello di avvio include un'AMI personalizzata, l'AMI deve soddisfare i requisiti in Specifica di un'AMI. Quando si aggiorna il gruppo di nodi a una versione più recente del modello di avvio, tutti i nodi vengono riciclati in modo da corrispondere alla nuova configurazione della versione del modello di avvio specificata.
> Non è possibile aggiornare direttamente un gruppo di nodi implementato senza un modello di avvio a una nuova versione del modello di avvio. È invece necessario implementare un nuovo gruppo di nodi utilizzando il modello di avvio per aggiornare il gruppo di nodi a una nuova versione del modello di avvio.

È possibile aggiornare un gruppo di nodi alla stessa versione della versione di Kubernetes del piano di controllo (control-plane). Ad esempio, se hai un cluster che esegue Kubernetes `1.29`, puoi aggiornare i nodi che attualmente eseguono Kubernetes `1.28` alla versione `1.29` con il comando seguente.

```
eksctl upgrade nodegroup \
  --name=node-group-name \
  --cluster=my-cluster \
  --region=region-code \
```

```
--kubernetes-version=1.29
```

AWS Management Console

Per aggiornare una versione di un gruppo di nodi con AWS Management Console

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegliere il cluster che contiene il gruppo di nodi da aggiornare.
3. Se almeno un gruppo di nodi dispone di un aggiornamento disponibile, nella parte superiore della pagina viene visualizzata una casella di notifica dell'aggiornamento disponibile. Se selezioni la scheda Compute (Calcolo), visualizzerai Update now (Aggiorna ora) nella colonna AMI release version (Versione rilascio AMI) nella tabella Node groups (Gruppi di nodi) per il gruppo di nodi per cui è disponibile un aggiornamento. Per aggiornare il gruppo di nodi, scegli Update now (Aggiorna ora).

   Non verrà visualizzata una notifica per i gruppi di nodi implementati con un'AMI personalizzata. Se i nodi vengono implementati con un'AMI personalizzata, completare la procedura seguente per implementare una nuova AMI personalizzata aggiornata.

   a. Crea una nuova versione dell'AMI.

   b. Creare una nuova versione del modello di avvio con il nuovo ID AMI.

   c. Aggiornamento dei i nodi alla nuova versione del modello di avvio.
4. Nella finestra di dialogo Update node group version (Aggiorna la versione del gruppo di nodi), attiva o disattiva le seguenti opzioni:

   - Update node group version (Aggiorna la versione del gruppo di nodi): questa opzione non è disponibile se hai implementato un'AMI personalizzata o se l'AMI ottimizzata per Amazon EKS attualmente è disponibile nella versione più recente del cluster.
   - Change launch template version (Modifica la versione del modello di avvio): questa opzione non è disponibile se il gruppo di nodi è implementato senza un modello di avvio personalizzato. È possibile aggiornare la versione del modello di avvio solo per un gruppo di nodi implementato con un modello di avvio personalizzato. Seleziona la Launch template version (Versione del modello di avvio) a cui eseguire l'aggiornamento del gruppo di nodi. Se il gruppo di nodi è configurato con un'AMI personalizzata, anche la versione selezionata deve specificare un'AMI. Quando si esegue l'aggiornamento a una versione più recente

del modello di avvio, tutti i nodi vengono riciclati in modo da corrispondere alla nuova configurazione della versione del modello di avvio specificata.

5. Per Update strategy (Strategia aggiornamento), seleziona una delle seguenti opzioni:

   - Aggiornamento in sequenza: questa opzione rispetta i budget di interruzione dei Pod per il cluster. Gli aggiornamenti non riescono se c'è un problema di budget che interrompe il Pod che fa sì che Amazon EKS non riesca a svuotare correttamente i Pods in esecuzione su questo gruppo di nodi.
   - Forza aggiornamento: questa opzione non rispetta i budget per le interruzioni dei Pod. Gli aggiornamenti vengono eseguiti indipendentemente dai problemi di budget di interruzione dei Pod forzando il riavvio dei nodi.

6. Scegli Aggiorna.

Modificare la configurazione di un gruppo di nodi

È possibile modificare parte della configurazione di un gruppo di nodi gestiti.

Per modificare la configurazione di un gruppo di nodi

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegliere il cluster che contiene il gruppo di nodi da modificare.
3. Seleziona la scheda Compute (Calcolo).
4. Seleziona il gruppo di nodi da modificare, e scegli Edit (Modifica).
5. (Facoltativo) Nella pagina Edit node group (Modifica gruppo di nodi), effettua le seguenti operazioni:

   a. Modifica la configurazione di scalabilità del gruppo di nodi.

      - Dimensione desiderata: specifica il numero corrente di nodi che il gruppo di nodi gestiti deve mantenere.
      - Dimensione minima: specifica il numero minimo di nodi a cui il gruppo di nodi gestiti può essere ridotto.
      - Dimensione massima: specifica il numero massimo di nodi a cui il gruppo di nodi gestiti può essere aumentato orizzontalmente. Per il numero massimo di nodi supportati in un gruppo di nodi, vedere Service Quotas di Amazon EKS.

b. (Facoltativo) Aggiungi o rimuovi etichette Kubernetes ai nodi nel gruppo di nodi. Le etichette mostrate qui sono solo le etichette applicate con Amazon EKS. Altre etichette, non mostrate qui, possono essere presenti sui nodi.

c. (Facoltativo) Aggiungi o rimuovi taint Kubernetes ai nodi nel gruppo di nodi. I taint aggiunti possono avere l'effetto di **`NoSchedule`**, **`NoExecute`**, oppure **`PreferNoSchedule`**. Per ulteriori informazioni, consulta Taint dei nodi nei gruppi di nodi gestiti.

d. (Facoltativo) Aggiungi o rimuovi Tag dalla risorsa del gruppo di nodi. Questi tag vengono applicati solo al gruppo di nodi Amazon EKS. I tag dei gruppi di nodi non si propagano ad altre risorse come istanze o le sottoreti di Amazon EC2 nel gruppo di nodi.

e. (Facoltativo) Modifica la Configurazione dell'aggiornamento del gruppo di nodi. Seleziona un'opzione tra Numero o Percentuale.

- Numero: seleziona e specifica il numero di nodi nel gruppo nodi che possono essere aggiornati in parallelo. Questi nodi non saranno disponibili durante l'aggiornamento.
- Percentuale: selezionare e specificare la percentuale di nodi nel gruppo di nodi che possono essere aggiornati in parallelo. Questi nodi non saranno disponibili durante l'aggiornamento. Questa funzione è utile se si dispone di diversi nodi nel gruppo di nodi.

f. Al termine della modifica, scegli Salva modifiche.

Comportamento dell'aggiornamento del nodo gestito

La strategia di aggiornamento del nodo (worker) gestito di Amazon EKS ha quattro fasi diverse descritte nelle sezioni seguenti.

Fase di configurazione

La fase di configurazione prevede i seguenti passaggi:

1. Crea una nuova versione del modello di avvio Amazon EC2 per il gruppo con scalabilità automatica associato al gruppo di nodi. La nuova versione del modello di avvio utilizza l'AMI di destinazione o una versione del modello di avvio personalizzata per l'aggiornamento.
2. Il gruppo con scalabilità automatica viene aggiornato per utilizzare la versione più recente del modello di avvio.
3. Determina la quantità massima di nodi da aggiornare in parallelo utilizzando la proprietà `updateConfig` per il gruppo di nodi. Il massimo non disponibile ha una quota di 100 nodi. Il

valore di default è un nodo. Per ulteriori informazioni, consultare la proprietà `updateConfig` nella Documentazione di riferimento delle API di Amazon EKS.

Fase di scalabilità

Quando si aggiornano i nodi in un gruppo di nodi gestiti, i nodi aggiornati vengono avviati nella stessa zona di disponibilità di quelli in fase di aggiornamento. Per garantire questo posizionamento, utilizziamo Availability Zone Rebalancing di Amazon EC2. Per ulteriori informazioni, consultare Availability Zone Rebalancing nella Guida per l'utente di Amazon EC2 Auto Scaling. Per soddisfare questo requisito, è possibile avviare fino a due istanze per zona di disponibilità nel gruppo di nodi gestiti.

La fase di scalabilità prevede i seguenti passaggi:

1. Incrementa la dimensione massima e la dimensione desiderata del gruppo con scalabilità automatica in base alla scelta maggiore tra:
   - Fino a due volte il numero di zone di disponibilità in cui è stato implementato il gruppo Auto Scaling.
   - Il massimo non disponibile per l'aggiornamento.

     Ad esempio, se il gruppo di nodi ha cinque zone di disponibilità e `maxUnavailable` è uno, il processo di aggiornamento può avviare al massimo 10 nodi. Tuttavia, quando `maxUnavailable` è 20 (o qualcosa superiore a 10), il processo lancerebbe 20 nuovi nodi.
2. Dopo il dimensionamento del gruppo con scalabilità automatica, controlla se i nodi che utilizzano la configurazione più recente sono presenti nel gruppo di nodi. Questo passaggio ha esito positivo solo quando soddisfa i seguenti criteri:
   - Almeno un nuovo nodo viene avviato in ogni zona di disponibilità in cui esiste il nodo.
   - Ogni nuovo nodo deve essere nello stato `Ready`.
   - I nuovi nodi devono avere etichette applicate da Amazon EKS.

     Queste sono le etichette applicate da Amazon EKS sui nodi (worker) di un normale gruppo di nodi:

     - `eks.amazonaws.com/nodegroup-image=`*`$amiName`*
     - `eks.amazonaws.com/nodegroup=`*`$nodeGroupName`*

     Queste sono le etichette applicate da Amazon EKS sui nodi (worker) in un modello di avvio personalizzato o un gruppo di nodi AMI:

- `eks.amazonaws.com/nodegroup-image=$amiName`
- `eks.amazonaws.com/nodegroup=$nodeGroupName`
- `eks.amazonaws.com/sourceLaunchTemplateId=$launchTemplateId`
- `eks.amazonaws.com/sourceLaunchTemplateVersion=$launchTemplateVersion`

3. Contrassegna i nodi come non programmabili per evitare di programmare nuovi Pods. Inoltre, etichetta i nodi con `node.kubernetes.io/exclude-from-external-load-balancers=true` per rimuovere i nodi dai sistemi di bilanciamento del carico prima di terminare i nodi.

Di seguito sono riportati i motivi noti che portano a un errore `NodeCreationFailure` in questa fase:

Capacità insufficiente nella zona di disponibilità

Esiste la possibilità che la zona di disponibilità non abbia capacità disponibile per i tipi di istanza richiesti. Si consiglia di configurare più tipi di istanze durante la creazione di un gruppo di nodi gestiti.

Limiti delle istanze EC2 dell'account

Potrebbe essere necessario aumentare il numero di istanze Amazon EC2 che il l'account può eseguire contemporaneamente utilizzando Service Quotas. Per ulteriori informazioni, consulta Service Quotas di EC2 nella Guida per l'utente di Amazon Elastic Compute Cloud per le istanze Linux.

Dati utente personalizzati

I dati utente personalizzati possono talvolta interrompere il processo di bootstrap. Questo scenario può portare al mancato avvio di `kubelet` sul nodo o sui nodi che non ricevono le etichette Amazon EKS previste. Per ulteriori informazioni, consulta Specifica di un'AMI.

Qualsiasi modifica che renda un nodo non integro o non pronto

La pressione del disco del nodo, la pressione della memoria e condizioni simili possono far sì che un nodo non assuma lo stato `Ready`.

Fase di aggiornamento

La fase di aggiornamento prevede i seguenti passaggi:

1. Seleziona casualmente un nodo che deve essere aggiornato, fino al massimo non disponibile configurato per il gruppo di nodi.
2. Svuota i Pods dal nodo. Se i Pods non lasciano il nodo entro 15 minuti e non c'è un flag di forzatura, la fase di aggiornamento non riesce con errore `PodEvictionFailure`. Per questo scenario, puoi applicare il flag di forzatura con la richiesta `update-nodegroup-version` di eliminare i Pods.
3. Isola il nodo dopo che ogni Pod è stato espulso e aspetta 60 secondi. Ciò consente al controller di servizio di non inviare nuove richieste a questo nodo, e lo rimuove dall'elenco di nodi attivi.
4. Invia una richiesta di interruzione al gruppo con scalabilità automatica per il nodo isolato.
5. Ripete i passaggi di aggiornamento precedenti fino a quando non vi sono nodi nel gruppo implementato con la versione precedente del modello di avvio.

Di seguito sono riportati i motivi noti che portano a un errore `PodEvictionFailure` in questa fase:

PDB aggressivo

Sul Pod è definito un PDB aggressivo oppure sono presenti più PDB che puntano allo stesso Pod.

Implementazione che tollera tutti i taint

Una volta espulso ogni Pod, il nodo dovrebbe essere vuoto in quanto oggetto di taint nei passaggi precedenti. Tuttavia, se l'implementazione tollera ogni taint, è più probabile che il nodo non sia vuoto, causando un errore di espulsione del Pod.

Fase di dimensionamento

La fase di dimensionamento diminuisce di uno la dimensione massima del gruppo Auto Scaling e la dimensione desiderata per tornare ai valori prima dell'avvio dell'aggiornamento.

Se il flusso di lavoro di aggiornamento determina che il Cluster Autoscaler stia dimensionando il gruppo di nodi durante la fase di dimensionamento del flusso di lavoro, esce immediatamente senza riportare il gruppo di nodi alle dimensioni originali.

Taint dei nodi nei gruppi di nodi gestiti

Amazon EKS supporta la configurazione di taint Kubernetes attraverso gruppi di nodi gestiti. I taint e le tolleranze agiscono insieme per garantire che i Pods non siano pianificati su nodi inappropriati. Ad un nodo possono essere applicati uno o più taint. Questo indica che il nodo non dovrebbe accettare

alcun Pods che non tollera i taint. Le tolleranze vengono applicate ai Pods e consentono, ma non forzano, la pianificazione dei Pods su nodi con taint corrispondenti. Per ulteriori informazioni consulta Taint e tolleranze nella documentazione di Kubernetes.

I taint dei nodi Kubernetes possono essere applicati a gruppi di nodi gestiti nuovi ed esistenti utilizzando la AWS Management Console o tramite l'API Amazon EKS.

- Per informazioni sulla creazione di un gruppo di nodi con un taint utilizzando la AWS Management Console, consulta Creazione di un gruppo di nodi gestiti.
- Di seguito è riportato un esempio di creazione di un gruppo di nodi con un taint utilizzando la AWS CLI:

```
aws eks create-nodegroup \
 --cli-input-json '
{
  "clusterName": "my-cluster",
  "nodegroupName": "node-taints-example",
  "subnets": [
     "subnet-1234567890abcdef0",
     "subnet-abcdef01234567890",
     "subnet-021345abcdef67890"
   ],
  "nodeRole": "arn:aws:iam::111122223333:role/AmazonEKSNodeRole",
  "taints": [
     {
         "key": "dedicated",
         "value": "gpuGroup",
         "effect": "NO_SCHEDULE"
     }
   ]
}'
```

Per ulteriori informazioni ed esempi di utilizzo, consulta taint nella documentazione di riferimento di Kubernetes.

> **Note**
>
> - I taint possono essere aggiornati dopo aver creato il gruppo di nodi utilizzando l'API `UpdateNodegroupConfig`.

- La chiave del taint deve iniziare con una lettera o un numero. Può includere lettere, numeri, trattini (`-`), punti (`.`) e trattini bassi (`_`). Può contenere fino a 63 caratteri.
- Facoltativamente, la chiave del taint può iniziare con un prefisso di sottodominio DNS e un singolo `/`. Se inizia con un prefisso di sottodominio DNS, può avere una lunghezza di 253 caratteri.
- Il valore è facoltativo e deve iniziare con una lettera o un numero. Può includere lettere, numeri, trattini (`-`), punti (`.`) e trattini bassi (`_`). Può contenere fino a 63 caratteri.
- Quando si utilizza Kubernetes direttamente o la AWS Management Console, l'effetto taint deve essere **`NoSchedule`**, **`PreferNoSchedule`** o **`NoExecute`**. Tuttavia, quando si utilizza AWS CLI direttamente o l'API, l'effetto taint deve essere **`NO_SCHEDULE`**, **`PREFER_NO_SCHEDULE`** o **`NO_EXECUTE`**.
- Sono consentiti un massimo di 50 taint per un gruppo di nodi.
- Se i taint creati utilizzando un gruppo di nodi gestito vengono rimossi manualmente da un nodo, Amazon EKS non li aggiunge nuovamente al nodo. Questo è vero anche se i taint sono specificati nella configurazione del gruppo di nodi gestito.

Puoi utilizzare il comando AWS CLI `aws eks update-nodegroup-config` per aggiungere, rimuovere o sostituire i taint per i gruppi di nodi gestiti.

Personalizzazione di nodi gestiti con un modello di avvio

Per ottenere il massimo livello di personalizzazione, è possibile implementare i nodi gestiti utilizzando il proprio modello di avvio. L'utilizzo di un modello di avvio consente funzionalità come le seguenti:

- Fornisci argomenti di bootstrap all'implementazione di un nodo, come ad esempio argomenti `kubelet` aggiuntivi.
- Assegna indirizzi IP ai Pods da un blocco CIDR diverso rispetto all'indirizzo IP assegnato al nodo.
- Implementa la tua AMI personalizzata sui nodi.
- Implementa il tuo CNI personalizzato sui nodi.

Quando fornisci il tuo modello di avvio alla prima creazione di un gruppo di nodi gestito, in seguito avrai anche una maggiore flessibilità. Dopo aver implementato un gruppo di nodi gestiti con un modello di avvio personalizzato, potrai aggiornarlo in maniera iterativa con una versione diversa dello stesso modello. Quando si aggiorna il gruppo di nodi a una versione diversa del modello di avvio, tutti

i nodi del gruppo vengono riciclati in modo da corrispondere alla nuova configurazione della versione del modello di avvio specificata.

I gruppi di nodi gestiti vengono sempre implementati con un modello di avvio da utilizzare con il gruppo di dimensionamento automatico Amazon EC2. Quando non viene fornito un modello di avvio, l'API Amazon EKS ne crea automaticamente uno con i valori predefiniti nel tuo account. Tuttavia, consigliamo di non modificare i modelli di avvio generati automaticamente. Inoltre, i gruppi di nodi esistenti che non utilizzano un modello di avvio personalizzato non possono essere aggiornati direttamente. A tale scopo, sarà invece necessario creare un nuovo gruppo di nodi con un modello di avvio personalizzato.

Informazioni di base sulla configurazione del modello di avvio

Puoi creare un modello di lancio di Amazon EC2 Auto Scaling con AWS CLI, o un AWS Management Console SDK. AWS Per ulteriori informazioni, consulta Creazione di un modello di avvio per un gruppo con scalabilità automatica nella Guida per l'utente di Dimensionamento automatico Amazon EC2. Alcune impostazioni di un modello di avvio sono simili a quelle utilizzate per la configurazione dei nodi gestiti. Quando si implementa o si aggiorna un gruppo di nodi con un modello di avvio, è necessario specificare alcune impostazioni nella configurazione del gruppo di nodi o nel modello di avvio. Non specificare un'impostazione in entrambe le posizioni. Se un'impostazione è settata dove non dovrebbe, operazioni come la creazione o l'aggiornamento di un gruppo di nodi non riescono.

Nella tabella seguente sono elencate le impostazioni vietate in un modello di avvio. Elenca inoltre impostazioni simili, se disponibili, che sono necessarie per la configurazione del gruppo di nodi gestiti. Le impostazioni elencate sono quelle visualizzate nella console. Potrebbero avere nomi simili ma diversi nell'SDK and. AWS CLI

Modello di avvio – Vietato	Configurazione del gruppo di nodi Amazon EKS
Sottorete in Interfacce di rete (Aggiungi interfaccia di rete)	Sottoreti in Configurazione di rete del gruppo di nodi nella pagina Specifica reti
Profilo dell'istanza IAM in Dettagli avanzati	Ruolo IAM del nodo in Configurazione del gruppo di nodi nella pagina Configura gruppo di nodi
Comportamento di arresto e Interrompi - Iberna comportamento in Dettagli avanzati. Mantieni default Non include nell'impostazione	Nessun equivalente. Amazon EKS deve controllare il ciclo di vita dell'istanza, non il gruppo Auto Scaling.

Modello di avvio – Vietato	Configurazione del gruppo di nodi Amazon EKS
del modello di avvio nel modello di avvio per entrambe le impostazioni.	

Nella tabella seguente sono elencate le impostazioni vietate nella configurazione di un gruppo di nodi gestiti. Elenca inoltre impostazioni simili, se disponibili, richieste in un modello di avvio. Le impostazioni elencate sono quelle visualizzate nella console. Potrebbero avere nomi simili in AWS CLI and SDK.

Configurazione del gruppo di nodi Amazon EKS – Vietata	Modello di avvio
(Solo se è stata specificata un'AMI personalizzata in un modello di avvio) Tipo di AMI in Configurazione di calcolo del gruppo di nodi sulla pagina Imposta configurazione di calcolo e dimensionamento: la console mostra il messaggio Specificato nel modello di lancio e l'ID AMI specificato. Se nel modello di avvio non è stato specificato Application and OS Images (Amazon Machine Image) (Applicazione e immagini OS [Amazon Machine Image]), è possibile selezionare un'AMI nella configurazione del gruppo di nodi.	Application and OS Images (Amazon Machine Image) (Applicazione e immagini OS [Amazon Machine Image]) in Launch template contents (Contenuti del modello di avvio): è necessario specificare un ID se soddisfi uno dei seguenti requisiti: • É in esecuzione un'AMI personalizzata. Se si specifica un'AMI che non soddisfa i requisiti elencati in Specifica di un'AMI, l'implementazione del gruppo di nodi non andrà a buon fine. • Necessità di fornire i dati utente per passare gli argomenti al file `bootstrap.sh` incluso nell'AMI ottimizzata per Amazon EKS. Puoi consentire alle istanze di assegnare un numero significativamente maggiore di indirizzi IPPods, assegnare indirizzi IP Pods da un blocco CIDR diverso da quello dell'istanza o implementare un cluster privato senza accesso a Internet in uscita. Per ulteriori informazioni, consulta i seguenti argomenti:

Configurazione del gruppo di nodi Amazon EKS – Vietata	Modello di avvio
	• Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2 • Rete personalizzata per i pod • Requisiti dei cluster privati • Specifica di un'AMI
Dimensioni del disco in Configurazione di calcolo del gruppo di nodi nella pagina Imposta configurazione di calcolo e dimensionamento: la console mostra il messaggio Specificato nel modello di lancio.	Dimensioni in Archiviazione (volumi) (Aggiungi nuovo volume). È necessario specificarlo nel modello di avvio.
Coppia di chiavi SSH in Configurazione del gruppo di nodi nella pagina Specifica reti: la console mostra la chiave specificata nel modello di avvio o mostra il messaggio Non specificato nel modello di lancio.	Nome della coppia di chiavi in Coppia di chiavi (login).
Non è possibile specificare gruppi di sicurezza di origine a cui è consentito l'accesso remoto quando si utilizza un modello di avvio.	Gruppi di sicurezza in Impostazioni di rete per l'istanza o Gruppi di sicurezza in Interfacce di rete (Aggiungi interfaccia di rete), ma non entrambi. Per ulteriori informazioni, consulta Utilizzo di gruppi di sicurezza personalizzati.

Note

- Se si esegue l'implementazione di un gruppo di nodi utilizzando un modello di avvio, specificare uno o nessun Tipo di istanza in Contenuti del modello di avvio in un modello di avvio. In alternativa, è possibile specificare 0-20 tipi di istanza per Tipi di istanza nella pagina Impostare la configurazione di calcolo e dimensionamento della console. In alternativa, è possibile farlo utilizzando altri strumenti che utilizzano l'API Amazon EKS. Se si specifica un tipo di istanza in un modello di avvio e si utilizza tale modello di avvio per implementare il gruppo di nodi, non è possibile specificare alcun tipo di istanza nella

console o utilizzare altri strumenti che sfruttino l'API Amazon EKS. Se non si specifica un tipo di istanza in un modello di avvio nella console o in altri strumenti che utilizzano l'API Amazon EKS, il tipo di istanza `t3.medium` verrà utilizzato. Se il gruppo di nodi utilizza il tipo di capacità Spot, è consigliabile specificare più tipi di istanza utilizzando la console. Per ulteriori informazioni, consulta Tipi di capacità del gruppo di nodi gestiti.

- Se i container che si implementano nel gruppo di nodi utilizzano Instance Metadata Service versione 2, assicurarsi di impostare il Limite hop di risposta metadati a 2 nel modello di avvio. Per ulteriori informazioni, consulta Metadati e dati dell'utente delle istanze nella Guida per l'utente di Amazon EC2. Se si implementa un gruppo di nodi gestito senza utilizzare un modello di avvio personalizzato, questo valore viene impostato automaticamente per il gruppo di nodi nel modello di avvio predefinito.

Assegnazione di tag a istanze Amazon EC2

É possibile utilizzare il parametro `TagSpecification` di un modello di avvio per specificare quali tag applicare alle istanze Amazon EC2 nel gruppo di nodi. L'entità IAM che chiama le API `CreateNodegroup` o `UpdateNodegroupVersion` devono disporre delle autorizzazioni per `ec2:RunInstances` e `ec2:CreateTags`, e i tag devono essere aggiunti al modello di avvio.

Utilizzo di gruppi di sicurezza personalizzati

É possibile utilizzare un modello di avvio per specificare i gruppi di sicurezza Amazon EC2 da applicare alle istanze del gruppo di nodi. Questo può essere nel parametro gruppi di sicurezza a livello di istanza o come parte dei parametri di configurazione dell'interfaccia di rete. Non è possibile avviare un'istanza da un modello di avvio che specifica sia il livello di istanza sia l'interfaccia di rete dei gruppi di sicurezza. Considerare le seguenti condizioni che si applicano all'utilizzo di gruppi di sicurezza personalizzati con gruppi di nodi gestiti:

- Amazon EKS consente solo modelli di avvio con una singola specifica di interfaccia di rete.
- Per impostazione predefinita, Amazon EKS applica il Gruppo di sicurezza del cluster alle istanze nel gruppo di nodi, per facilitare la comunicazione tra i nodi e il piano di controllo. Se si specificano gruppi di sicurezza personalizzati nel modello di avvio utilizzando entrambe le opzioni menzionate in precedenza, Amazon EKS non aggiunge il gruppo di sicurezza del cluster. Pertanto, è necessario assicurarsi che le regole in entrata e in uscita dei gruppi di sicurezza abilitino la comunicazione con l'endpoint del cluster. Se le regole del gruppo di sicurezza non sono corrette, i nodi (worker) non possono aggiungersi al cluster. Per ulteriori informazioni sulle regole del gruppo di sicurezza, consultare Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS.

- Se è necessario l'accesso SSH alle istanze nel gruppo di nodi, includere un gruppo di sicurezza che consenta tale accesso.

Dati utente Amazon EC2

Il modello di avvio include una sezione per i dati utente personalizzati. È possibile specificare le impostazioni di configurazione per il gruppo di nodi in questa sezione senza creare manualmente singole AMI personalizzate. Per ulteriori informazioni sulle impostazioni disponibili per Bottlerocket, consulta Utilizzo dei dati utente su GitHub.

É possibile fornire i dati utente Amazon EC2 nel tuo modello di avvio utilizzando `cloud-init` durante l'avvio delle istanze. Per ulteriori informazioni, consultare la documentazione cloud-init. I dati utente possono essere utilizzati per eseguire operazioni di configurazione comuni. Sono comprese le seguenti opzioni:

- Inclusione di utenti o gruppi.
- Installazione di pacchetti

I dati utente di Amazon EC2 nei modelli di avvio utilizzati con i gruppi di nodi gestiti devono essere nel formato Archivio MIME in più parti per le AMI Amazon Linux e nel formato TOML per le AMI Bottlerocket. Questo perché i dati utente vengono uniti con i dati utente Amazon EKS necessari ai nodi per aderire al cluster. Non specificare alcun comando nei dati utente che avviano o modificano `kubelet`. Questa operazione viene eseguita come parte dei dati utente uniti da Amazon EKS. Certi parametri `kubelet`, come l'impostazione delle etichette sui nodi, possono essere configurati direttamente tramite l'API dei gruppi di nodi gestiti.

> **Note**
>
> Per ulteriori informazioni sulle personalizzazioni `kubelet` avanzate, tra cui l'avvio manuale o il passaggio a parametri di configurazione personalizzati, vedere Specifica di un'AMI. Amazon EKS non unisce i dati utente se un ID AMI viene specificato in un modello di avvio.

I seguenti dettagli forniscono ulteriori informazioni sulla sezione dei dati utente.

Amazon Linux 2 user data

È possibile unire più blocchi di dati utente in un unico blocco, detto file MIME in più parti. Ad esempio, puoi combinare un hook di avvio del cloud per la configurazione del daemon Docker con uno script di shell dei dati utente che installa un pacchetto personalizzato. Un file MIME in più parti è composto dai seguenti elementi:

- Il tipo di contenuto e la dichiarazione di delimitazione della parte: `Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="`
- La dichiarazione della versione MIME: `MIME-Version: 1.0`
- Uno o più blocchi di dati utente, che contengono i seguenti elementi:
  - La delimitazione di apertura, che indica l'inizio di un blocco di dati utente: `--==MYBOUNDARY==`
  - La dichiarazione del tipo di contenuto per il blocco: `Content-Type: text/cloud-config; charset="us-ascii"`. Per ulteriori informazioni sui tipi di contenuto, consultare la documentazione di cloud-init.
  - Il contenuto dei dati utente, ad esempio un elenco di comandi shell o direttive `cloud-init`.
  - La delimitazione di chiusura, che indica la fine del file MIME in più parti: `--==MYBOUNDARY==--`

Di seguito è riportato un esempio di un file in più parti MIME che è possibile utilizzare per crearne uno.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
echo "Running custom user data script"

--==MYBOUNDARY==--
```

Amazon Linux 2023 user data

Amazon Linux 2023 (AL2023) introduce un nuovo processo di inizializzazione dei nodi `nodeadm` che utilizza uno schema di configurazione YAML. Se utilizzi gruppi di nodi autogestiti o un'AMI

con un modello di avvio, ora dovrai fornire esplicitamente metadati del cluster aggiuntivi quando crei un nuovo gruppo di nodi. Un esempio dei parametri minimi richiesti è il seguente `apiServerEndpointcertificateAuthority`, dove ora `cidr` sono richiesti e il servizio:

```
---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig
spec:
  cluster:
    name: my-cluster
    apiServerEndpoint: https://example.com
    certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
    cidr: 10.100.0.0/16
```

In genere questa configurazione viene impostata nei dati utente, così com'è o incorporata in un documento MIME composto da più parti:

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="BOUNDARY"

--BOUNDARY
Content-Type: application/node.eks.aws

---
apiVersion: node.eks.aws/v1alpha1
kind: NodeConfig spec: [...]

--BOUNDARY--
```

In AL2, i metadati di questi parametri sono stati rilevati dalla chiamata `DescribeCluster` API di Amazon EKS. Con AL2023, questo comportamento è cambiato poiché la chiamata API aggiuntiva rischia di rallentare durante l'up-up di nodi su larga scala. Questa modifica non ha effetto su di te se utilizzi gruppi di nodi gestiti senza un modello di avvio o se utilizzi. Karpenter Per ulteriori informazioni sul `certificateAuthority` servizio`cidr`, consulta `DescribeCluster` la pagina Amazon EKS API Reference.

Bottlerocket user data

Bottlerocket struttura i dati utente nel formato TOML. È possibile fornire i dati utente da unire con i dati utente forniti da Amazon EKS. Ad esempio, è possibile fornire ulteriori impostazioni `kubelet`.

```
[settings.kubernetes.system-reserved]
```

```
cpu = "10m"
memory = "100Mi"
ephemeral-storage= "1Gi"
```

Per ulteriori informazioni sulle impostazioni supportate, consulta la documentazione Bottlerocket. È possibile configurare etichette di nodi e taint nei dati utente. Consigliamo, tuttavia, di configurarle all'interno del gruppo di nodi. In questo caso, Amazon EKS applica queste configurazioni.

Quando i dati utente vengono uniti, la formattazione non viene mantenuta, ma il contenuto rimane invariato. La configurazione fornita nei dati utente sostituisce tutte le impostazioni configurate da Amazon EKS. Quindi, se imposti `settings.kubernetes.max-pods` o`settings.kubernetes.cluster-dns-ip`, questi valori nei dati utente vengono applicati ai nodi.

Amazon EKS non supporta tutti i TOML validi. Di seguito è riportato un elenco di formati noti non supportati:

- Quote all'interno delle chiavi stimate: `'quoted "value"' = "value"`
- Quote evase nei valori: `str = "I'm a string. \"You can quote me\""`
- Galleggianti e numeri interi misti: `numbers = [ 0.1, 0.2, 0.5, 1, 2, 5 ]`
- Tipi misti nelle matrici: `contributors = ["foo@example.com", { name = "Baz", email = "baz@example.com" }]`
- Intestazioni tra parentesi con chiavi stimate: `[foo."bar.baz"]`

Windows user data

I dati utente di Windows utilizzano comandi PowerShell. Quando crei un gruppo di nodi gestiti, i dati utente personalizzati si combinano con i dati utente gestiti da Amazon EKS. I tuoi comandi PowerShell hanno la priorità, seguiti dai comandi dei dati utente gestiti, tutti all'interno di un unico tag **<powershell></powershell>**.

> **Note**
>
> Quando non è specificato alcun ID AMI nel modello di avvio, nei dati utente non utilizzare lo script bootstrap di Amazon EKS per Windows per configurare Amazon EKS.

Di seguito sono riportati dati utente di esempio.

```
<powershell>
Write-Host "Running custom user data script"
</powershell>
```

Specifica di un'AMI

Se si dispone di uno dei seguenti requisiti, specificare un ID AMI nel campo `ImageId` del modello di avvio. Selezionare il requisito di cui si dispone per ulteriori informazioni.

Fornire i dati utente per passare argomenti al file **`bootstrap.sh`** incluso nell'AMI Linux/Bottlerocket ottimizzata per Amazon EKS

Il termine "bootstrapping" indica l'aggiunta di comandi che possono essere eseguiti all'avvio di un'istanza. Ad esempio, il bootstrap consente di utilizzare argomenti `kubelet` aggiuntivi. È possibile passare gli argomenti allo script `bootstrap.sh` utilizzando `eksctl` senza specificare un modello di avvio. Oppure è possibile farlo specificando le informazioni nella sezione dati utente di un modello di avvio.

eksctl without specifying a launch template

Crea un file denominato *`my-nodegroup.yaml`* con i seguenti contenuti. Sostituisci ogni *`example value`* con i valori in tuo possesso. Gli argomenti `--apiserver-endpoint`, `--b64-cluster-ca` e `--dns-cluster-ip` sono facoltativi, ma grazie alla loro definizione lo script `bootstrap.sh` può evitare di effettuare una chiamata a `describeCluster`. Ciò è utile nelle configurazioni di cluster privati o nei cluster in cui si effettuano dimensionamenti frequenti dei nodi. Per ulteriori informazioni sullo script `bootstrap.sh`, consulta il file `bootstrap.sh` su GitHub.

- L'unico argomento richiesto è il nome del cluster (*`my-cluster`*).
- Per recuperare l'ID di un'AMI ottimizzata per `ami-`*`1234567890abcdef0`*, puoi fare riferimento alle tabelle nelle sezioni seguenti:
  - Recupero ID delle AMI Amazon Linux ottimizzate per Amazon EKS
  - Recupero degli ID AMI Bottlerocket ottimizzate per Amazon EKS
  - Recupero degli ID AMI Windows ottimizzate per Amazon EKS
- Per recuperare il valore *`certificate-authority`* per il cluster, esegui il comando seguente.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
 --name my-cluster --region region-code
```

- Per recuperare il valore *api-server-endpoint* per il cluster, esegui il comando seguente.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
cluster --region region-code
```

- Il valore per `--dns-cluster-ip` è il tuo servizio CIDR con `.10` alla fine. Per recuperare il valore *service-cidr* per il cluster, esegui il comando seguente. Ad esempio, se il valore restituito è `ipv4 10.100.0.0/16`, il tuo valore è *10.100.0.10*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
 --output text --name my-cluster --region region-code
```

- Questo esempio fornisce un argomento `kubelet` per impostare un valore `max-pods` personalizzato utilizzando lo script `bootstrap.sh` incluso nell'AMI ottimizzata per Amazon EKS. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura. Per assistenza nella scelta di *my-max-pods-value*, consulta Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods.

```
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code

managedNodeGroups:
  - name: my-nodegroup
    ami: ami-1234567890abcdef0
    instanceType: m5.large
    privateNetworking: true
    disableIMDSv1: true
    labels: { x86-al2-specified-mng }
    overrideBootstrapCommand: |
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster \
```

```
        --b64-cluster-ca certificate-authority \
        --apiserver-endpoint api-server-endpoint \
        --dns-cluster-ip service-cidr.10 \
        --kubelet-extra-args '--max-pods=my-max-pods-value' \
        --use-max-pods false
```

Per ogni opzione disponibile per il file `eksctl config`, consulta Schema del file config nella documentazione su `eksctl`. L'utility `eksctl` crea autonomamente un modello di avvio e popola i dati utente con i dati forniti nel file `config`.

Crea un gruppo di nodi con il comando seguente.

```
eksctl create nodegroup --config-file=my-nodegroup.yaml
```

User data in a launch template

Specifica le seguenti informazioni nella sezione dati utente del modello di avvio. Sostituisci ogni *example value* con i valori in tuo possesso. Gli argomenti `--apiserver-endpoint`, `--b64-cluster-ca` e `--dns-cluster-ip` sono facoltativi, ma grazie alla loro definizione lo script `bootstrap.sh` può evitare di effettuare una chiamata a `describeCluster`. Ciò è utile nelle configurazioni di cluster privati o nei cluster in cui si effettuano dimensionamenti frequenti dei nodi. Per ulteriori informazioni sullo script `bootstrap.sh`, consulta il file `bootstrap.sh` su GitHub.

- L'unico argomento richiesto è il nome del cluster (*my-cluster*).
- Per recuperare il valore *certificate-authority* per il cluster, esegui il comando seguente.

  ```
  aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text
   --name my-cluster --region region-code
  ```

- Per recuperare il valore *api-server-endpoint* per il cluster, esegui il comando seguente.

  ```
  aws eks describe-cluster --query "cluster.endpoint" --output text --name my-
  cluster --region region-code
  ```

- Il valore per `--dns-cluster-ip` è il tuo servizio CIDR con `.10` alla fine. Per recuperare il valore *service-cidr* per il cluster, esegui il comando seguente. Ad esempio, se il valore restituito è `ipv4 10.100.0.0/16`, il tuo valore è *10.100.0.10*.

  ```
  aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr"
   --output text --name my-cluster --region region-code
  ```

- Questo esempio fornisce un argomento `kubelet` per impostare un valore `max-pods` personalizzato utilizzando lo script `bootstrap.sh` incluso nell'AMI ottimizzata per Amazon EKS. Per assistenza nella scelta di *`my-max-pods-value`*, consulta Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
set -ex
/etc/eks/bootstrap.sh my-cluster \
  --b64-cluster-ca certificate-authority \
  --apiserver-endpoint api-server-endpoint \
  --dns-cluster-ip service-cidr.10 \
  --kubelet-extra-args '--max-pods=my-max-pods-value' \
  --use-max-pods false

--==MYBOUNDARY==--
```

Fornire i dati utente per passare argomenti al file **`Start-EKSBootstrap.ps1`** incluso nell'AMI Windows ottimizzata per Amazon EKS

Il termine "bootstrapping" indica l'aggiunta di comandi che possono essere eseguiti all'avvio di un'istanza. È possibile passare gli argomenti allo script `Start-EKSBootstrap.ps1` utilizzando `eksctl` senza specificare un modello di avvio. Oppure è possibile farlo specificando le informazioni nella sezione dati utente di un modello di avvio.

Per specificare un ID AMI Windows personalizzato, fai le seguenti considerazioni:

- Devi utilizzare un modello di avvio e fornire i comandi bootstrap richiesti nella sezione dei dati utente. Per recuperare l'ID Windows desiderato, puoi utilizzare la tabella in AMI Windows ottimizzate per Amazon EKS.
- Esistono vari limiti e condizioni. Ad esempio, è necessario aggiungere `eks:kube-proxy-windows` alla mappa di configurazione di AWS IAM Authenticator. Per ulteriori informazioni, consulta Limiti e condizioni quando si specifica un ID AMI.

Specifica le seguenti informazioni nella sezione dati utente del modello di avvio. Sostituisci ogni *example value* con i valori in tuo possesso. Gli argomenti `-APIServerEndpoint`, `-Base64ClusterCA` e `-DNSClusterIP` sono facoltativi, ma grazie alla loro definizione lo script `Start-EKSBootstrap.ps1` può evitare di effettuare una chiamata a `describeCluster`.

- L'unico argomento richiesto è il nome del cluster (*my-cluster*).
- Per recuperare il valore *certificate-authority* per il cluster, esegui il comando seguente.

```
aws eks describe-cluster --query "cluster.certificateAuthority.data" --output text --
name my-cluster --region region-code
```

- Per recuperare il valore *api-server-endpoint* per il cluster, esegui il comando seguente.

```
aws eks describe-cluster --query "cluster.endpoint" --output text --name my-cluster
 --region region-code
```

- Il valore per `--dns-cluster-ip` è il tuo servizio CIDR con `.10` alla fine. Per recuperare il valore *service-cidr* per il cluster, esegui il comando seguente. Ad esempio, se il valore restituito è `ipv4 10.100.0.0/16`, il tuo valore è *10.100.0.10*.

```
aws eks describe-cluster --query "cluster.kubernetesNetworkConfig.serviceIpv4Cidr" --
output text --name my-cluster --region region-code
```

- Per ulteriori argomenti, consulta.Parametri di configurazione dello script di bootstrap.

> **Note**
>
> Se utilizzi il servizio personalizzato CIDR, devi specificarlo utilizzando il parametro `-ServiceCIDR`. In caso contrario, la risoluzione DNS per i Pods nel cluster fallirà.

```
<powershell>
[string]$EKSBootstrapScriptFile = "$env:ProgramFiles\Amazon\EKS\Start-EKSBootstrap.ps1"
& $EKSBootstrapScriptFile -EKSClusterName my-cluster `
  -Base64ClusterCA certificate-authority `
  -APIServerEndpoint api-server-endpoint `
  -DNSClusterIP service-cidr.10
</powershell>
```

Eseguire un'AMI personalizzata a causa di specifici requisiti di sicurezza, conformità o policy interne

Per ulteriori informazioni, consultare Amazon Machine Images (AMI) nella Guida per l'utente di Amazon EC2. La specifica di build dell'AMI Amazon EKS contiene risorse e script di configurazione per creare un'AMI Amazon EKS personalizzata basata su Amazon Linux. Per ulteriori informazioni, consulta Specifica della build AMI di Amazon EKS su GitHub. Per creare AMI personalizzate installate con altri sistemi operativi, consulta AMI personalizzate di esempio di Amazon EKS su GitHub.

> **Important**
>
> Quando si specifica una AMI, Amazon EKS non unisce alcun dato utente. É l'utente, piuttosto, ad essere responsabile della fornitura dei comandi `bootstrap` richiesti per unire i nodi al cluster. Se i nodi non riescono a unirsi al cluster, le operazioni `CreateNodegroup` e `UpdateNodegroupVersion` di Amazon EKS non vengono eseguite con successo.

Limiti e condizioni quando si specifica un ID AMI

Di seguito sono riportati i limiti e le condizioni per la specifica di un ID AMI con gruppi di nodi gestiti:

- È necessario creare un nuovo gruppo di nodi per passare dalla specifica o meno di un ID AMI in un modello di avvio.
- Non si riceve una notifica nella console quando è disponibile una versione AMI più recente. Per aggiornare il gruppo di nodi a una versione AMI più recente, devi creare una nuova versione del modello di avvio con un ID AMI aggiornato. Quindi, è necessario aggiornare il gruppo di nodi con la nuova versione del modello di avvio.
- I seguenti campi non possono essere impostati nell'API se si specifica un ID AMI:
  - `amiType`
  - `releaseVersion`
  - `version`
- Se si specifica un ID AMI, qualsiasi set di `taints` nell'API viene applicato in modo asincrono. Per applicare i taint prima che un nodo si unisca al cluster, è necessario passare i taint a `kubelet` utilizzando il flag della riga di comando `--register-with-taints`. Per ulteriori informazioni, consulta `kubelet` nella documentazione Kubernetes.

- Quando specifichi un ID AMI personalizzato per i gruppi di nodi Windows gestiti, aggiungilo `eks:kube-proxy-windows` alla mappa di configurazione di AWS IAM Authenticator. Questa API è necessaria per il funzionamento di DNS.

  1. Apri la mappa di configurazione di AWS IAM Authenticator per modificarla.

     ```
     kubectl edit -n kube-system cm aws-auth
     ```

  2. Aggiungi questa voce all'elenco `groups` sotto ogni `rolearn` associato ai nodi Windows. La tua mappa di configurazione dovrebbe avere un aspetto simile a [aws-auth-cm-windows.yaml].

     ```
     - eks:kube-proxy-windows
     ```

  3. Salva il file ed esci dall'editor di testo.

Eliminazione di un gruppo di nodi gestiti

In questo argomento viene descritto come eliminare un gruppo di nodi gestiti Amazon EKS. Quando elimini un gruppo di nodi gestiti, Amazon EKS imposterà su zero la dimensione minima, massima e desiderata del tuo gruppo Auto Scaling. Questo fa sì che il gruppo di nodi venga dimensionato.

Prima di arrestare ogni istanza, Amazon EKS invia un segnale per svuotare i Pods da quel nodo. Se i Pods non vengono svuotati dopo alcuni minuti, Amazon EKS lascia che Auto Scaling continui l'interruzione dell'istanza. Una volta terminate tutte le istanze, il gruppo Auto Scaling viene eliminato.

> **Important**
>
> Se si elimina un gruppo di nodi gestiti che utilizza un ruolo IAM del nodo non impiegato da nessun altro gruppo di nodi gestiti nel cluster, il ruolo viene rimosso da `aws-auth ConfigMap`. Se i gruppi di nodi autogestiti nel cluster utilizzano lo stesso ruolo IAM del nodo, i nodi autogestiti passeranno allo stato `NotReady`. Inoltre, anche l'operazione del cluster viene interrotta. Per aggiungere una mappatura per il ruolo che stai utilizzando solo per i gruppi di nodi autogestiti, consulta la sezione Creazione di voci di accesso, se la versione della piattaforma del tuo cluster corrisponde almeno alla versione minima elencata nella sezione dei prerequisiti di Gestire le voci di accesso. Se la tua versione della piattaforma è precedente alla versione minima richiesta per le voci di accesso, puoi

aggiungere nuovamente la voce a `aws-auth` `ConfigMap`. Per ulteriori informazioni, digita `eksctl create iamidentitymapping --help` nel terminale.

È possibile creare un gruppo di nodi gestiti con `eksctl` o la AWS Management Console.

eksctl

Per eliminare un gruppo di nodi gestiti con **`eksctl`**

Inserire il seguente comando. Sostituisci ogni *`example value`* con i valori in tuo possesso.

```
eksctl delete nodegroup \
  --cluster my-cluster \
  --name my-mng \
  --region region-code
```

Per ulteriori opzioni, consulta Eliminazione e svuotamento dei gruppi di nodi nella documentazione di `eksctl`.

AWS Management Console

Per eliminare il gruppo di nodi gestito con AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nella pagina Cluster, scegli il cluster che contiene il gruppo di nodi da eliminare.
3. Nella pagina del cluster selezionata, scegli la scheda Calcolo.
4. Nella sezione Gruppi di nodi, scegliere il gruppo di nodi da eliminare. Scegli Elimina.
5. Nella finestra di dialogo di conferma di Eliminazione del gruppo di nodi, inserisci il nome del gruppo di nodi. Scegli Elimina.

AWS CLI

Per eliminare il gruppo di nodi gestito con AWS CLI

1. Inserire il seguente comando. Sostituisci ogni *`example value`* con i valori in tuo possesso.

```
aws eks delete-nodegroup \
```

```
--cluster-name my-cluster \
--nodegroup-name my-mng \
--region region-code
```

2. Usa i tasti freccia sulla tastiera per scorrere l'output di risposta. Premi il tasto **q** quando hai finito.

Per ulteriori informazioni, consulta il comando `delete-nodegroup` in Documentazione di riferimento sui comandi di AWS CLI .

Nodi autogestiti

Un cluster contiene uno o più nodi Amazon EC2 su cui sono pianificati i Pods. I nodi Amazon EKS vengono eseguiti nel tuo AWS account e si connettono al piano di controllo del cluster tramite l'endpoint del server API del cluster. La loro fatturazione è basata sui prezzi Amazon EC2. Per ulteriori informazioni, consulta Prezzi di Amazon EC2.

Un cluster può contenere diversi gruppi di nodi. Ciascun gruppo di nodi contiene uno o più nodo implementati in un gruppo Amazon EC2 Auto Scaling. Il tipo di istanza dei nodi all'interno del gruppo può variare, ad esempio quando si utilizza la selezione del tipo di istanza basata sugli attributi con Karpenter. Tutte le istanze di un gruppo di nodi devono utilizzare il ruolo IAM del nodo Amazon EKS.

Amazon EKS fornisce Amazon Machine Image (AMI) specializzate chiamate AMI ottimizzate per Amazon EKS. Le AMI sono configurate per funzionare con Amazon EKS. I loro componenti includono `containerd` e AWS IAM Authenticator. `kubelet` L'AMI contiene anche uno script di bootstrap specializzato che consente di individuare e connettersi automaticamente al piano di controllo del cluster.

Se si limita l'accesso all'endpoint pubblico del cluster utilizzando blocchi CIDR, è consigliabile abilitare anche l'accesso agli endpoint privati. In questo modo i nodi possono comunicare con il cluster. Senza l'endpoint privato abilitato, i blocchi CIDR specificati per l'accesso pubblico devono includere le origini di uscita dal VPC. Per ulteriori informazioni, consulta Controllo accessi all'endpoint del cluster Amazon EKS.

Per aggiungere nodi autogestiti al cluster Amazon EKS, consultare gli argomenti che seguono. Se si avviano manualmente i nodi autogestiti, è necessario aggiungere il seguente tag a ciascun nodo. Per ulteriori informazioni, consultare Aggiunta ed eliminazione di tag in una singola risorsa. Se si seguono i passaggi della guida, il tag richiesto viene aggiunto al nodo per conto dell'utente.

Chiave	Valore
`kubernetes.io/cluster/ my-cluster`	`owned`

Per ulteriori informazioni sui nodi da un prospettiva Kubernetes generale, consulta Nodi nella documentazione Kubernetes.

Argomenti

- Avvio di nodi Amazon Linux autogestiti
- Avvio dei nodi Bottlerocket autogestiti
- Avvio dei nodi Windows autogestiti
- Avvio dei nodi Ubuntu autogestiti
- Aggiornamenti del nodo autogestito

Avvio di nodi Amazon Linux autogestiti

In questo argomento viene descritto come avviare gruppi con scalabilità automatica di nodi Linux che si registrano con il cluster Amazon EKS. Dopo che i nodi vengono aggiunti al cluster, puoi implementare le applicazioni Kubernetes per gli stessi. Puoi anche avviare nodi Amazon Linux autogestiti con `eksctl` o. AWS Management Console Se devi avviare nodi su AWS Outposts, consultaAvvio di nodi Amazon Linux autogestiti su un Outpost.

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Creazione di un cluster Amazon EKS. Se hai delle sottoreti nel luogo in Regione AWS cui hai AWS Outposts AWS Wavelength, o le AWS Local Zones sono abilitate, tali sottoreti non devono essere state passate al momento della creazione del cluster.
- Un ruolo IAM esistente per i nodi da utilizzare. Per crearne uno, consulta Ruolo IAM del nodo Amazon EKS. Se questo ruolo non prevede nessuna delle policy per VPC CNI, per i pod VPC CNI è necessario il ruolo separato riportato di seguito.
- (Facoltativo, ma consigliato) Il componente aggiuntivo Amazon VPC CNI plugin for Kubernetes configurato con il suo personale ruolo IAM a cui è allegata la policy IAM necessaria. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

- Familiarità con le considerazioni riportate in Scelta di un tipo di istanza Amazon EC2. A seconda del tipo di istanza scelto, potrebbero esserci ulteriori prerequisiti per il cluster e il VPC.

eksctl

> **Note**
>
> `eksctl`al momento non supporta Amazon Linux 2023.

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Avvio di nodi Linux autogestiti tramite **`eksctl`**

1. (Facoltativo) Se la policy IAM gestita AmazonEKS_CNI_Policy è collegata al Ruolo IAM del nodo Amazon EKS, consigliamo, invece, di assegnarla a un ruolo IAM associato all'account di servizio Kubernetes `aws-node`. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).
2. Il comando seguente crea un gruppo di nodi in un cluster esistente. Sostituisci *`al-nodes`* con un nome per il gruppo di nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura. Sostituisci *`my-cluster`* con il nome del cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può contenere più di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster. Sostituisci i *`example value`* rimanenti con i valori in tuo possesso. Per impostazione predefinita, i nodi vengono creati con la stessa versione Kubernetes del piano di controllo.

   Prima di scegliere un valore per `--node-type`, verifica Scelta di un tipo di istanza Amazon EC2.

Sostituisci *my-key* con il nome della coppia di chiavi Amazon EC2 o della chiave pubblica. Questa chiave viene utilizzata per eseguire il SSH nei nodi dopo il loro avvio. Se non hai già una coppia di chiavi Amazon EC2, puoi crearla nella AWS Management Console. Per ulteriori informazioni, consulta la sezione relativa alle coppie di chiavi Amazon EC2 nella Guida per l'utente di Amazon EC2.

Crea il tuo gruppo di nodi con il comando seguente.

> **Important**
>
> Se desideri distribuire un gruppo di nodi su sottoreti Wavelength o Local AWS Outposts Zone, ci sono altre considerazioni:
>
> - Le sottoreti non devono essere state trasmesse al momento della creazione del cluster.
> - È necessario creare il gruppo di nodi con un file di configurazione, specificando le sottoreti e `volumeType: gp2`. Per ulteriori informazioni, consulta Creazione di un gruppo di nodi da un file di configurazione e lo Schema del file config nella documentazione di `eksctl`.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --name al-nodes \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --ssh-access \
  --managed=false \
  --ssh-public-key my-key
```

Per implementare un gruppo di nodi che:

- può assegnare un numero significativamente più elevato di indirizzi IP a Pods rispetto alla configurazione predefinita, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2.

- può assegnare gli indirizzi `IPv4` a Pods da un altro blocco CIDR rispetto a quello dell'istanza, consulta Rete personalizzata per i pod.
- può assegnare indirizzi `IPv6` a Pods e servizi, consulta `IPv6`indirizzi per cluster Pods e services.
- utilizza il runtime `containerd`, devi implementare il gruppo di nodi utilizzando un file di `config`. Per ulteriori informazioni, consulta Prova la migrazione da Docker a `containerd`.
- non dispone di accesso a Internet in uscita, consulta Requisiti dei cluster privati.

Per un elenco completo di tutte le opzioni e i valori predefiniti disponibili, immetti il comando seguente.

```
eksctl create nodegroup --help
```

Se i nodi non riescono a unirsi al cluster, consulta Impossibile aggiungere i nodi al cluster nella Guida alla risoluzione dei problemi.

Di seguito viene riportato un output di esempio: Durante la creazione dei nodi vengono generate diverse righe. Una delle ultime righe di output è simile alla seguente riga di esempio.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facoltativo) Implementa un'applicazione di esempio per testare il cluster e i nodi Linux.
4. Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

   - Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
   - No, Pods nel cluster è richiesto l'accesso al servizio di metadati dell'istanza Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

   Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.

AWS Management Console

Fase 1: Avvio di nodi Linux autogestiti tramite la AWS Management Console

1. Scarica l'ultima versione del modello. AWS CloudFormation

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
   ```

2. Attendi che lo stato del cluster risulti `ACTIVE`. Se vengono avviati prima che il cluster sia attivo, i nodi non riescono a effettuare la registrazione al cluster e sarà necessario riavviarli.
3. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.
4. Scegli Crea stack e quindi seleziona Con nuove risorse (standard).
5. In Specifica modello, seleziona Carica un file di modello e Scegli file.
6. Seleziona il file `amazon-eks-nodegroup.yaml` scaricato.
7. Seleziona Successivo.
8. Nella pagina Specifica i dettagli dello stack, immetti i parametri seguenti e scegli Successivo:

   - Nome stack: scegli il nome per lo stack di AWS CloudFormation . Ad esempio, è possibile chiamarlo ***my-cluster-nodes***. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può contenere più di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
   - ClusterName: inserisci il nome che hai usato per creare il cluster Amazon EKS. Questo nome deve corrispondere al nome del cluster o i nodi non verranno aggiunti al cluster.
   - ClusterControlPlaneSecurityGruppo: scegli il SecurityGroupsvalore dall' AWS CloudFormation output che hai generato quando hai creato il tuo VPC.

     Nella procedura seguente viene illustrata un'operazione per recuperare il gruppo applicabile.

     1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
     2. Scegli il nome del cluster.
     3. Scegli la scheda Reti.

4. Utilizza il valore dei gruppi di sicurezza aggiuntivi come riferimento quando selezioni dall'elenco a discesa ClusterControlPlaneSecurityGruppo.

- NodeGroupNome: inserisci un nome per il tuo gruppo di nodi. Questo nome può essere utilizzato in seguito per identificare il gruppo di nodi con dimensionamento automatico creato per i tuoi nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura.
- NodeAutoScalingGroupMinSize: Inserisci il numero minimo di nodi su cui il gruppo Auto Scaling del nodo può scalare.
- NodeAutoScalingGroupDesiredCapacity: Inserisci il numero di nodi desiderato su cui scalare quando viene creato lo stack.
- NodeAutoScalingGroupMaxSize: Inserisci il numero massimo di nodi su cui il gruppo Auto Scaling del nodo può scalare orizzontalmente.
- NodeInstanceTipo: scegli un tipo di istanza per i tuoi nodi. Per ulteriori informazioni, consulta Scelta di un tipo di istanza Amazon EC2.
- NodeImageIDSSMParam: precompilato con il parametro Amazon EC2 Systems Manager di una recente AMI ottimizzata per Amazon EKS per una versione variabile. Kubernetes Per utilizzare una versione secondaria diversa di Kubernetes supportata da Amazon EKS, sostituisci *1.XX* con una versione supportata differente. Si consiglia di specificare la stessa versione Kubernetes del cluster.

  Puoi anche sostituirlo *amazon-linux-2* con un altro tipo di AMI. Per ulteriori informazioni, consulta Recupero ID delle AMI Amazon Linux ottimizzate per Amazon EKS.

  > **Note**
  >
  > L'AMI del nodo Amazon EKS è basata su Amazon Linux. Tieni traccia degli eventi di sicurezza o di privacy per Amazon Linux 2 nel Centro di sicurezza di Amazon Linux o iscriviti al feed RSS associato. Gli eventi di sicurezza e privacy includono una panoramica del problema, quali sono i pacchetti interessati e come aggiornare le istanze per risolvere il problema.

- NodeImageID: (Facoltativo) Se utilizzi la tua AMI personalizzata (anziché l'AMI ottimizzata per Amazon EKS), inserisci un ID AMI del nodo per il tuo Regione AWS. Se specifichi un valore qui, questo sostituisce tutti i valori nel campo NodeImageIDSsmParam.
- NodeVolumeDimensione: specifica la dimensione del volume root per i tuoi nodi, in GiB.

- NodeVolumeTipo: specifica un tipo di volume root per i tuoi nodi.
- KeyName: inserisci il nome di una coppia di chiavi SSH Amazon EC2 che puoi usare per connetterti tramite SSH ai tuoi nodi dopo il loro avvio. Se non disponi di una coppia di chiavi Amazon EC2, puoi crearla nella AWS Management Console. Per ulteriori informazioni, consulta la sezione relativa alle coppie di chiavi Amazon EC2 nella Guida per l'utente di Amazon EC2.

  > **Note**
  >
  > Se non fornisci una key pair qui, la creazione dello AWS CloudFormation stack fallisce.

- BootstrapArguments: Specificate eventuali argomenti opzionali da passare allo script di bootstrap del nodo, ad esempio argomenti aggiuntivi`kubelet`. Per ulteriori informazioni, leggi le informazioni sull'utilizzo dello script di bootstrap su GitHub.

  Per implementare un gruppo di nodi che:

  - può assegnare un numero significativamente più elevato di indirizzi IP a Pods rispetto alla configurazione predefinita, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2.
  - può assegnare gli indirizzi `IPv4` a Pods da un altro blocco CIDR rispetto a quello dell'istanza, consulta Rete personalizzata per i pod.
  - può assegnare indirizzi `IPv6` a Pods e servizi, consulta `IPv6`indirizzi per cluster Pods e services.
  - utilizza il runtime `containerd`, devi implementare il gruppo di nodi utilizzando un file di `config`. Per ulteriori informazioni, consulta Prova la migrazione da Docker a `containerd`.
  - non dispone di accesso a Internet in uscita, consulta Requisiti dei cluster privati.

- DisableIMDSv1: ogni nodo supporta Instance Metadat Service versione 1 (IMDSv1) e IMDSv2 per impostazione predefinita. Puoi disabilitare IMDSv1. Per evitare che in futuro i nodi e i Pods nel gruppo di nodi utilizzino MDSv1, imposta DisableIMDSv1 su true. Per ulteriori informazioni su IMDS, consulta Configurazione del servizio di metadati dell'istanza. Per ulteriori informazioni sulle relative limitazioni dell'accesso ai nodi, consulta Limita l'accesso al profilo dell'istanza assegnato al nodo (worker).
- VpcId: inserisci l'ID per il VPC che hai creato.

- Sottoreti: scegli le sottoreti create per il VPC. Se il VPC è stato creato utilizzando i passaggi descritti in Creazione di un VPC per il cluster Amazon EKS, specificare solo le sottoreti private all'interno del VPC per avviare i nodi. È possibile visualizzare le sottoreti private aprendo ogni collegamento relativo alla sottorete dalla scheda Reti del cluster.

  > **Important**
  >
  > - Se una qualsiasi delle sottoreti è pubblica, devi abilitare l'impostazione di assegnazione automatica degli indirizzi IP pubblici. Se l'impostazione non è abilitata per la sottorete pubblica, a tutti i nodi distribuiti in quella sottorete pubblica non verrà assegnato un indirizzo IP pubblico e non saranno in grado di comunicare con il cluster o altri servizi. AWS Se la sottorete è stata distribuita prima del 26 marzo 2020 utilizzando uno dei modelli AWS CloudFormation VPC di Amazon EKS o utilizzando`eksctl`, l'assegnazione automatica degli indirizzi IP pubblici è disabilitata per le sottoreti pubbliche. Per informazioni su come abilitare l'assegnazione di indirizzi IP pubblici per una sottorete, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv4` pubblici per la sottorete. Se il nodo viene distribuito in una sottorete privata, è in grado di comunicare con il cluster e altri servizi tramite un gateway NAT. AWS
  > - Se le sottoreti non hanno accesso a Internet, leggi con attenzione le considerazioni e le fasi aggiuntive descritte in Requisiti dei cluster privati.
  > - Se si AWS Outposts selezionano le sottoreti Wavelength o Local Zone, le sottoreti non devono essere state passate al momento della creazione del cluster.

9. Seleziona le opzioni desiderate nella pagina Configura opzioni dello stack, quindi scegli Next (Avanti).
10. Seleziona la casella di controllo a sinistra di I acknowledge that AWS CloudFormation might create IAM resources (Riconosco che CFN potrebbe creare risorse IAM), quindi scegli Create stack (Crea stack).
11. Al termine della creazione dello stack, selezionalo nella console e scegli Output.
12. Registra il NodeInstanceruolo per il gruppo di nodi che è stato creato. Ciò sarà utile quando configurerai i nodi di Amazon EKS.

Fase 2: abilitazione dell'aggiunta di nodi al cluster

> **Note**
>
> Se i nodi sono stati avviati all'interno di un VPC privato senza accesso a Internet in uscita, assicurati di abilitare i nodi da aggiungere al cluster dal VPC.

1. Verifica se disponi già di una `ConfigMap` per `aws-auth`.

   ```
   kubectl describe configmap -n kube-system aws-auth
   ```

2. Se ti viene mostrata una `ConfigMap` per `aws-auth`, aggiornala se necessario.

   a. Apri `ConfigMap` per la modifica.

   ```
   kubectl edit -n kube-system configmap/aws-auth
   ```

   b. Aggiungi una nuova voce `mapRoles`, se necessario. Impostate il `rolearn` valore sul valore del NodeInstanceruolo che avete registrato nella procedura precedente.

   ```
   [...]
   data:
     mapRoles: |
       - rolearn: <ARN of instance role (not instance profile)>
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
   [...]
   ```

   c. Salva il file ed esci dall'editor di testo.

3. Se hai ricevuto un messaggio di errore che indica "`Error from server (NotFound): configmaps "aws-auth" not found`", applica lo `ConfigMap` di stock.

   a. Scarica la mappa di configurazione.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/aws-auth-cm.yaml
   ```

b. Nel `aws-auth-cm.yaml` file, impostate il `rolearn` valore sul valore del NodeInstanceruolo registrato nella procedura precedente. Per eseguire questa operazione, utilizza un editor di testo o sostituisci *`my-node-instance-role`* eseguendo il comando seguente:

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-instance-role|' aws-auth-cm.yaml
```

c. Applica la configurazione. L'esecuzione di questo comando potrebbe richiedere alcuni minuti.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Guarda lo stato dei nodi e attendi che raggiungano lo stato `Ready`.

```
kubectl get nodes --watch
```

Inserisci `Ctrl+C` per tornare a un prompt dello shell (interprete di comandi).

> **Note**
>
> Se ricevi qualsiasi altro errore di tipo di risorsa o autorizzazione, consulta la sezione Accesso negato o non autorizzato (`kubectl`) nell'argomento relativo alla risoluzione dei problemi.

Se i nodi di lavoro non riescono a unirsi al cluster, consulta Impossibile aggiungere i nodi al cluster nella Guida alla risoluzione dei problemi.

5. (Solo nodi GPU) Se hai scelto un tipo di istanza GPU e l'AMI accelerata ottimizzata per Amazon EKS, devi applicare il plug-in del dispositivo NVIDIA per Kubernetes come DaemonSet sul cluster con il comando seguente. Sostituisci *`vX.X.X`* con la versione plugin per dispositivi NVIDIA/K8Sdesiderata prima di eseguire il seguente comando.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/nvidia-device-plugin.yml
```

Fase 3: operazioni aggiuntive

1. (Facoltativo) Implementa un'applicazione di esempio per testare il cluster e i nodi Linux.
2. (Facoltativo) Se la policy IAM gestita AmazonEKS_CNI_Policy (se hai un cluster `IPv4`) o la policy *`AmazonEKS_CNI_IPv6_Policy`* (da te creata) se hai un cluster `IPv6`) è collegata al the section called “Ruolo IAM del nodo”, ti consigliamo invece di assegnarla invece a un ruolo IAM associato all'account del servizio `aws-node` Kubernetes. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).
3. Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

   - Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
   - No, Pods nel cluster è richiesto l'accesso al servizio di metadati dell'istanza Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

   Per ulteriori informazioni, consulta Limita l'accesso al profilo dell'istanza assegnato al nodo (worker).

Blocchi di capacità per ML

> **Important**
>
> - I Blocchi di capacità sono disponibili solo per determinati tipi di istanze Amazon EC2 e. Regioni AWS Per informazioni sulla compatibilità, consulta Work with Capacity Blocks Prerequisites nella Guida per l'utente di Amazon EC2 per le istanze Linux.
> - Al momento i Capacity Blocks non possono essere utilizzati con gruppi di nodi gestiti da Amazon EKS oKarpenter.

I blocchi di capacità per il machine learning (ML) ti consentono di riservare istanze GPU in una data futura per supportare i carichi di lavoro ML di breve durata. Le istanze eseguite all'interno di un Capacity Block vengono automaticamente posizionate vicine tra loro all'interno di Amazon UltraClusters EC2, quindi non è necessario utilizzare un gruppo di posizionamento del cluster. Per ulteriori informazioni, consulta Capacity Blocks for ML nella Guida per l'utente di Amazon EC2 per le istanze Linux.

Puoi utilizzare i blocchi di capacità con Amazon EKS per effettuare il provisioning e il dimensionamento dei nodi autogestiti. Le seguenti fasi forniscono una panoramica generale di esempio.

1. Crea un modello di avvio nella AWS Management Console. Per ulteriori informazioni, consulta Use Capacity Blocks per carichi di lavoro di machine learning nella Guida per l'utente di Amazon EC2 Auto Scaling.

   Assicurati di includere la configurazione del tipo di istanza e dell'Amazon Machine Image (AMI).
2. Collega il blocco di capacità a un modello di avvio utilizzando l'ID di prenotazione della capacità.

   Di seguito è riportato un modello di esempio per creare un AWS CloudFormation modello di lancio destinato a un Capacity Block:

   ```
   NodeLaunchTemplate:
       Type: "AWS::EC2::LaunchTemplate"
       Properties:
         LaunchTemplateData:
           InstanceMarketOptions:
             MarketType: "capacity-block"
           CapacityReservationSpecification:
             CapacityReservationTarget:
               CapacityReservationId: "cr-02168da1478b509e0"
           IamInstanceProfile:
             Arn: iam-instance-profile-arn
           ImageId: image-id
           InstanceType: p5.48xlarge
           KeyName: key-name
           SecurityGroupIds:
           - sg-05b1d815d1EXAMPLE
           UserData: user-data
   ```

   Devi trasmettere la sottorete nella zona di disponibilità in cui viene effettuata la prenotazione, perché i blocchi di capacità sono zonali.
3. Se stai creando il gruppo di nodi autogestito prima che la prenotazione della capacità diventi attiva, imposta la capacità desiderata su `0`. Quando crei il gruppo di nodi, assicurati di specificare solo la rispettiva sottorete per la zona di disponibilità in cui è la capacità è riservata.

   Di seguito è riportato un CloudFormation modello di esempio che può essere utilizzato. Questo esempio ottiene i valori di `LaunchTemplateId` e `Version` della risorsa `AWS::Amazon`

`EC2::LaunchTemplate` mostrata nell'esempio precedente. Ottiene anche i valori di `DesiredCapacity`, `MaxSize`, `MinSize` e `VPCZoneIdentifier` che sono dichiarati altrove nello stesso modello.

```
NodeGroup:
  Type: "AWS::AutoScaling::AutoScalingGroup"
  Properties:
    DesiredCapacity: !Ref NodeAutoScalingGroupDesiredCapacity
    LaunchTemplate:
      LaunchTemplateId: !Ref NodeLaunchTemplate
      Version: !GetAtt NodeLaunchTemplate.LatestVersionNumber
    MaxSize: !Ref NodeAutoScalingGroupMaxSize
    MinSize: !Ref NodeAutoScalingGroupMinSize
    VPCZoneIdentifier: !Ref Subnets
    Tags:
      - Key: Name
        PropagateAtLaunch: true
        Value: !Sub ${ClusterName}-${NodeGroupName}-Node
      - Key: !Sub kubernetes.io/cluster/${ClusterName}
        PropagateAtLaunch: true
        Value: owned
```

4. Una volta che il gruppo di nodi è stato creato, assicurati di registrare il valore `NodeInstanceRole` per il gruppo di nodi creato. Ciò è necessario per verificare che, in seguito al dimensionamento del gruppo di nodi, i nuovi nodi vengano aggiunti al cluster e Kubernetes sia in grado di riconoscerli. Per ulteriori informazioni, consulta le istruzioni della AWS Management Console in Avvio di nodi Amazon Linux autogestiti.

5. Consigliamo di creare una policy di dimensionamento programmato per il gruppo con dimensionamento automatico che si allinei ai tempi di prenotazione dei blocchi di capacità. Per ulteriori informazioni, consulta Dimensionamento programmato per Dimensionamento automatico Amazon EC2 nella Guida per l'utente di Dimensionamento automatico Amazon EC2.

   Puoi utilizzare tutte le istanze prenotate fino a 30 minuti prima dell'orario di fine del blocco di capacità. Le istanze ancora in esecuzione in quel momento inizieranno a terminare. Per avere tempo sufficiente per svuotare correttamente i nodi, suggeriamo di pianificare il dimensionamento fino a zero più di 30 minuti prima dell'orario di fine della prenotazione del blocco di capacità.

   Se desideri invece aumentare le dimensioni manualmente ogni volta che la prenotazione della capacità diventa `Active`, devi aggiornare la capacità desiderata del gruppo

con dimensionamento automatico all'inizio della prenotazione del blocco di capacità. Successivamente, dovrai anche ridurre le dimensioni manualmente più di 30 minuti prima dell'orario di fine della prenotazione del blocco di capacità.

6. Il gruppo di nodi è ora pronto per la pianificazione di carichi di lavoro e Pods.
7. Affinché tu possa Pods essere svuotato con eleganza, ti consigliamo di configurare AWS Node Termination Handler. Questo gestore sarà in grado di monitorare gli eventi del ciclo di vita «ASG Scale-in» di Amazon EC2 Auto EventBridge Scaling utilizzando e Kubernetes consentire al piano di controllo di intraprendere l'azione richiesta prima che l'istanza diventi non disponibile. Altrimenti, i tuoi oggetti Pods e Kubernetes rimarranno bloccati nello stato in sospeso. Per ulteriori informazioni, consulta Node Termination Handler su.AWS GitHub

   Se non configuri Node Termination Handler, consigliamo di cominciare a svuotare manualmente i Pods prima di raggiungere la finestra di 30 minuti, in modo che abbiano tempo sufficiente per il corretto svuotamento.

Avvio dei nodi Bottlerocket autogestiti

> **Note**
>
> I gruppi di nodi gestiti potrebbero offrire alcuni vantaggi per il tuo caso d'uso. Per ulteriori informazioni, consulta Gruppi di nodi gestiti.

Questo argomento descrive come avviare gruppi Auto Scaling di nodi Bottlerocket registrati nel cluster Amazon EKS. Bottlerocketè un sistema operativo open source Linux basato AWS che puoi utilizzare per eseguire container su macchine virtuali o host bare metal. Dopo che i nodi vengono aggiunti al cluster, puoi implementare le applicazioni Kubernetes per gli stessi. Per ulteriori informazioni su Bottlerocket, consulta Utilizzo di un'AMI Bottlerocket con Amazon EKS su GitHub e Supporto di AMI personalizzate nella documentazione di `eksctl`.

Per informazioni sugli aggiornamenti sul posto, consulta Operatore aggiornamenti di Bottlerocket su GitHub.

Important

- I nodi Amazon EKS sono istanze Amazon EC2 standard e la loro fatturazione è basata sui normali prezzi dell'istanza Amazon EC2. Per ulteriori informazioni, consulta Prezzi di Amazon EC2.
- Puoi avviare nodi Bottlerocket nei cluster estesi di Amazon EKS su AWS Outposts, ma non puoi avviarli in cluster locali su Outposts. AWS Per ulteriori informazioni, consulta Amazon EKS su AWS Outposts.
- È possibile eseguire l'implementazione su istanze Amazon EC2 con processori x86 o Arm. Tuttavia, non è possibile eseguire l'implementazione su istanze con chip Inferentia.
- Bottlerocketè compatibile con. AWS CloudFormation Tuttavia, non esiste un CloudFormation modello ufficiale che possa essere copiato per distribuire Bottlerocket nodi per Amazon EKS.
- Le immagini Bottlerocket non vengono fornite con una shell o un server SSH. Puoi utilizzare metodi di out-of-band accesso per consentire l'SSHattivazione del contenitore di amministrazione e per eseguire alcuni passaggi di configurazione di bootstrap con i dati utente. Per ulteriori informazioni, consulta le seguenti sezioni nel README.md di Bottlerocket su GitHub:
  - Esplorazione
  - Container amministratore
  - Impostazioni di Kubernetes

Per avviare i nodi Bottlerocket utilizzando **`eksctl`**

Questa procedura richiede `eksctl` versione `0.183.0` o successiva. Puoi verificare la versione con il comando seguente:

```
eksctl version
```

Per istruzioni sull'installazione o sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

> **Note**
>
> Questa procedura funziona solo per i cluster creati con `eksctl`.

1. Copia i seguenti contenuti sul dispositivo. Sostituisci *`my-cluster`* con il nome del cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster. Sostituisci *`ng-bottlerocket`* con un nome per il gruppo di nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura. Per implementare su istanze Arm, sostituire *`m5.large`* con un tipo di istanza Arm. Sostituisci *`my-ec2-keypair-name`* con il nome di una coppia di chiavi SSH Amazon EC2; che puoi utilizzare per connetterti utilizzando SSH nei nodi dopo l'avvio. Se non disponi di una coppia di chiavi Amazon EC2, puoi crearla nella AWS Management Console. Per ulteriori informazioni, consulta la sezione relativa alle coppie di chiavi Amazon EC2 nella Guida per l'utente di Amazon EC2. Sostituisci tutti i *`example values`* rimanenti con i valori in tuo possesso. Dopo aver effettuato le sostituzioni, eseguire il comando modificato per creare `bottlerocket.yaml`.

   Se si specifica un tipo di istanza Amazon EC2 Arm, esaminare le considerazioni in AMI Amazon Linux Arm ottimizzate per Amazon EKS prima dell'implementazione. Per istruzioni su come eseguire l'implementazione utilizzando un'AMI personalizzata, consulta Creazione di Bottlerocket su GitHub e Supporto per AMI personalizzate nella documentazione di `eksctl`. Per implementare un gruppo di nodi gestito, implementare un'AMI personalizzata utilizzando un modello di avvio. Per ulteriori informazioni, consulta Personalizzazione di nodi gestiti con un modello di avvio.

   > **Important**
   >
   > Per distribuire un gruppo di nodi nelle sottoreti della zona AWS locale AWS Outposts AWS Wavelength, don't pass AWS Outposts AWS Wavelength, o nelle sottoreti della zona AWS locale al momento della creazione del cluster. È necessario specificare le sottoreti nell'esempio seguente. Per ulteriori informazioni, vedere Creare un gruppo di nodi da un file di configurazione e lo Schema del file config nella documentazione su `eksctl`. Sostituiscilo *`region-code`* con quello in cui si trova il Regione AWS cluster.

```
cat >bottlerocket.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: '1.30'

iam:
  withOIDC: true

nodeGroups:
  - name: ng-bottlerocket
    instanceType: m5.large
    desiredCapacity: 3
    amiFamily: Bottlerocket
    ami: auto-ssm
    iam:
       attachPolicyARNs:
          - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

2. Implementare i nodi mediante il comando seguente.

```
eksctl create nodegroup --config-file=bottlerocket.yaml
```

Di seguito viene riportato un output di esempio:

Durante la creazione dei nodi vengono generate diverse righe. Una delle ultime righe di output è simile alla seguente riga di esempio.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facoltativo) Crea un volume persistente Kubernetes su un nodo Bottlerocket usando il plug-in CSI di Amazon EBS. Il driver Amazon EBS di default si basa su strumenti di file system non inclusi in Bottlerocket. Per ulteriori informazioni sulla creazione di una classe di archiviazione utilizzando il driver, consultare Driver CSI per Amazon EBS.
4. (Facoltativo) Per impostazione predefinita, `kube-proxy` imposta il parametro kernel `nf_conntrack_max` su un valore predefinito che può differire da quello che Bottlerocket imposta originariamente all'avvio. Per mantenere l'impostazione di default di Bottlerocket, modificare la configurazione `kube-proxy` con il comando seguente.

   ```
   kubectl edit -n kube-system daemonset kube-proxy
   ```

   Aggiungi `--conntrack-max-per-core` e `--conntrack-min` agli argomenti `kube-proxy` nell'esempio seguente. Un'impostazione di `0` non implica alcun cambiamento.

   ```
         containers:
         - command:
           - kube-proxy
           - --v=2
           - --config=/var/lib/kube-proxy-config/config
           - --conntrack-max-per-core=0
           - --conntrack-min=0
   ```

5. (Facoltativo) implementa un'applicazione di esempio per testare i nodi Bottlerocket.
6. Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

   - Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
   - Nessuno Pods nel cluster richiede l'accesso al servizio di metadati delle istanze Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

   Per ulteriori informazioni, consulta Limita l'accesso al profilo dell'istanza assegnato al nodo (worker).

Avvio dei nodi Windows autogestiti

In questo argomento viene descritto come avviare gruppi con scalabilità automatica di nodi Windows che si registrano con il cluster Amazon EKS. Dopo che i nodi vengono aggiunti al cluster, puoi implementare le applicazioni Kubernetes per gli stessi.

Important

- I nodi Amazon EKS sono istanze Amazon EC2 standard e la loro fatturazione è basata sui normali prezzi dell'istanza Amazon EC2. Per ulteriori informazioni, consulta Prezzi di Amazon EC2.
- Puoi avviare nodi Windows nei cluster estesi di Amazon EKS su AWS Outposts, ma non puoi avviarli in cluster AWS locali su Outposts. Per ulteriori informazioni, consulta Amazon EKS su AWS Outposts.

Abilita il supporto Windows per il cluster. Consigliamo di rivedere le considerazioni importanti prima di avviare un gruppo di nodi Windows. Per ulteriori informazioni, consulta Abilitazione del supporto Windows.

Puoi avviare i nodi Windows autogestiti con `eksctl` o la AWS Management Console.

eksctl

Avvio di nodiWindows Linux autogestiti tramite **`eksctl`**

Questa procedura presuppone che `eksctl` sia installato e che la versione `eksctl` sia almeno `0.183.0`. È possibile verificare la tua versione con il seguente comando.

```
eksctl version
```

Per istruzioni sull'installazione o sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Note

Questa procedura funziona solo per i cluster creati con `eksctl`.

1. (Facoltativo) Se la policy IAM gestita AmazonEKS_CNI_Policy (se hai un cluster IPv4) o la policy *AmazonEKS_CNI_IPv6_Policy* (da te creata) se hai un cluster IPv6) è collegata al the section called “Ruolo IAM del nodo”, ti consigliamo invece di assegnarla invece a un ruolo IAM associato all'account del servizio aws-node Kubernetes. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).
2. In questa procedura si presuppone la disponibilità di un cluster. Se non disponi già di un cluster Amazon EKS e di un gruppo di nodi Amazon Linux a cui aggiungere un gruppo di Windows nodi, ti consigliamo di seguire la Guida introduttiva ad Amazon EKS: eksctl guida. Le guide forniscono una spiegazione passo per passo completa per la creazione di un cluster Amazon EKS con nodi Amazon Linux.

   Crea il tuo gruppo di nodi con il comando seguente. Sostituiscilo *region-code* con Regione AWS quello in cui si trova il cluster. Sostituisci *my-cluster* con il nome del cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster. Sostituisci *ng-windows* con un nome per il gruppo di nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura. Per Kubernetes versione 1.24 o successiva, puoi sostituire *2019* con 2022 in modo da utilizzare Windows Server 2022. Sostituisci il resto di *example values* con i valori in tuo possesso.

   > **Important**
   >
   > Per distribuire un gruppo di nodi su o su sottoreti AWS Local AWS Outposts Zone, non passare le sottoreti AWS Outposts Wavelength o Local Zone quando crei il cluster. AWS Wavelength Crea il gruppo di nodi con un file di configurazione, specificando le sottoreti AWS Outposts Wavelength o Local Zone. Per ulteriori informazioni, vedere Creazione di un gruppo di nodi da un file di configurazione e lo Schema del file config nella documentazione su eksctl.

```
eksctl create nodegroup \
    --region region-code \
    --cluster my-cluster \
    --name ng-windows \
```

```
    --node-type t2.large \
    --nodes 3 \
    --nodes-min 1 \
    --nodes-max 4 \
    --managed=false \
    --node-ami-family WindowsServer2019FullContainer
```

> **Note**
>
> - Se i nodi non riescono a unirsi al cluster, consultare Impossibile aggiungere i nodi al cluster nella Guida alla risoluzione dei problemi.
> - Per vedere le opzioni disponibili per i comandi `eksctl`, inserire il comando seguente.
>
>   ```
>   eksctl command -help
>   ```

Di seguito viene riportato un output di esempio: Durante la creazione dei nodi vengono generate diverse righe. Una delle ultime righe di output è simile alla seguente riga di esempio.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facoltativo) Implementa un'applicazione di esempio per testare il cluster e i nodi Windows.
4. Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

   - Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
   - No, Pods nel cluster è richiesto l'accesso al servizio di metadati dell'istanza Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

   Per ulteriori informazioni, consulta Limita l'accesso al profilo dell'istanza assegnato al nodo (worker).

AWS Management Console

Prerequisiti

- Un cluster Amazon EKS esistente e un gruppo di nodi Linux. Se non disponi di tali risorse, ti consigliamo di seguire una delle nostre guide Guida introduttiva ad Amazon EKS per crearle. Le guide descrivono come creare un cluster Amazon EKS con nodi Linux.
- Creazione di un VPC e di un gruppo di sicurezza esistenti che soddisfano i requisiti per un cluster Amazon EKS. Per ulteriori informazioni, consulta Requisiti e considerazioni su VPC e sottoreti di Amazon EKS e Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS. La guida Guida introduttiva ad Amazon EKS consente di creare un VPC che soddisfa i requisiti. In alternativa, è possibile anche seguire Creazione di un VPC per il cluster Amazon EKS per crearne uno nuovo manualmente.
- Un cluster Amazon EKS esistente che utilizza un VPC e un gruppo di sicurezza in grado di soddisfare i requisiti di un cluster Amazon EKS. Per ulteriori informazioni, consulta Creazione di un cluster Amazon EKS. Se hai delle sottoreti nel luogo in Regione AWS cui hai AWS Outposts AWS Wavelength, o le AWS Local Zones sono abilitate, tali sottoreti non devono essere state passate al momento della creazione del cluster.

Fase 1: Avviare nodi autogestiti utilizzando WindowsAWS Management Console

1. Attendi che lo stato del cluster risulti `ACTIVE`. Se i nodi vengono avviati prima che il cluster sia attivo, la registrazione al cluster non riesce e sarà necessario riavviarli.
2. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation
3. Seleziona Crea stack.
4. Per Specifica modello, selezionare URL Amazon S3.
5. Copia il seguente URL e incollalo nell'URL di Amazon S3.

   ```
   https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2023-02-09/amazon-eks-windows-nodegroup.yaml
   ```

6. Seleziona due volte Next (Avanti).
7. Nella pagina Quick create stack (Creazione rapida pila), inserire i seguenti parametri di conseguenza:

- Nome stack: scegli il nome per lo stack di AWS CloudFormation . Ad esempio, è possibile chiamarlo ***my-cluster*-nodes**.
- ClusterName: inserisci il nome che hai usato per creare il cluster Amazon EKS.

  > **Important**
  >
  > Il nome deve corrispondere esattamente a quello presente nel Fase 1: Creazione del cluster Amazon EKS;. In caso contrario, i nodi non possono aggiungersi al cluster.

- ClusterControlPlaneSecurityGruppo: scegli il gruppo di sicurezza dall' AWS CloudFormation output che hai generato quando hai creato il tuo VPC.

  Nella procedura seguente viene illustrato un metodo per recuperare il gruppo applicabile.

  1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
  2. Scegli il nome del cluster.
  3. Scegli la scheda Reti.
  4. Utilizza il valore dei gruppi di sicurezza aggiuntivi come riferimento quando selezioni dall'elenco a discesa ClusterControlPlaneSecurityGruppo.
- NodeGroupNome: inserisci un nome per il tuo gruppo di nodi. Questo nome può essere utilizzato in seguito per identificare il gruppo di nodi con dimensionamento automatico creato per i tuoi nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura.
- NodeAutoScalingGroupMinSize: Inserisci il numero minimo di nodi su cui il gruppo Auto Scaling del nodo può scalare.
- NodeAutoScalingGroupDesiredCapacity: Inserisci il numero di nodi desiderato su cui scalare quando viene creato lo stack.
- NodeAutoScalingGroupMaxSize: Inserisci il numero massimo di nodi su cui il gruppo Auto Scaling del nodo può scalare orizzontalmente.
- NodeInstanceTipo: scegli un tipo di istanza per i tuoi nodi. Per ulteriori informazioni, consulta Scelta di un tipo di istanza Amazon EC2.

> **Note**
>
> I tipi di istanze supportati per la versione più recente del Amazon VPC CNI plugin for Kubernetes sono riportati in vpc_ip_resource_limit.go su GitHub. Potrebbe essere necessario aggiornare la versione CNI per utilizzare i tipi di istanze supportati più recenti. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.

- NodeImageIDSSMParam: precompilato con il parametro Amazon EC2 Systems Manager dell'attuale ID Core AMI ottimizzato per Amazon EKS. Windows Per utilizzare la versione completa di Windows, sostituisci *Core* con `Full`.
- NodeImageID: (Facoltativo) Se utilizzi la tua AMI personalizzata (anziché l'AMI ottimizzata per Amazon EKS), inserisci un ID AMI del nodo per il tuo Regione AWS. Se specifichi un valore per questo campo, questo sostituisce tutti i valori nel campo NodeImageIDSSMParam.
- NodeVolumeDimensione: specifica la dimensione del volume root per i tuoi nodi, in GiB.
- KeyName: inserisci il nome di una coppia di chiavi SSH Amazon EC2 che puoi usare per connetterti tramite SSH ai tuoi nodi dopo il loro avvio. Se non disponi di una coppia di chiavi Amazon EC2, puoi crearla nella AWS Management Console. Per ulteriori informazioni, consulta la sezione relativa alle coppie di chiavi Amazon EC2 nella Guida per l'utente di Amazon EC2.

> **Note**
>
> Se non fornisci una key pair qui, lo AWS CloudFormation stack non viene creato.

- BootstrapArguments: Specificate eventuali argomenti opzionali da passare allo script di bootstrap del nodo, come ad esempio l'utilizzo di `kubelet` argomenti aggiuntivi. `-KubeletExtraArgs`
- DisableIMDSv1: ogni nodo supporta Instance Metadat Service versione 1 (IMDSv1) e IMDSv2 per impostazione predefinita. Puoi disabilitare IMDSv1. Per evitare che in futuro i nodi e i Pods nel gruppo di nodi utilizzino MDSv1, imposta DisableIMDSv1 su true. Per ulteriori informazioni su IMDS, consulta Configurazione del servizio di metadati dell'istanza.
- VpcId: Seleziona l'ID per il VPC che hai creato.

- NodeSecurityGruppi: seleziona il gruppo di sicurezza che è stato creato per il tuo gruppo di Linux nodi quando hai creato il tuo VPC. Se ai nodi Linux è collegato più di un gruppo di sicurezza, specificarli tutti. Questo, ad esempio, se il gruppo di nodi Linux è stato creato con `eksctl`.
- Sottoreti: scegliere le sottoreti create. Se il VPC è stato creato utilizzando i passaggi descritti in Creazione di un VPC per il cluster Amazon EKS, specificare solo le sottoreti private all'interno del VPC per avviare i nodi.

> **Important**
>
> - Se una qualsiasi delle sottoreti è pubblica, devi abilitare l'impostazione di assegnazione automatica degli indirizzi IP pubblici. Se l'impostazione non è abilitata per la sottorete pubblica, ai nodi distribuiti in quella sottorete pubblica non verrà assegnato un indirizzo IP pubblico e non saranno in grado di comunicare con il cluster o altri servizi. AWS Se la sottorete è stata distribuita prima del 26 marzo 2020 utilizzando uno dei modelli AWS CloudFormation VPC di Amazon EKS o utilizzando`eksctl`, l'assegnazione automatica degli indirizzi IP pubblici è disabilitata per le sottoreti pubbliche. Per informazioni su come abilitare l'assegnazione di indirizzi IP pubblici per una sottorete, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv4` pubblici per la sottorete. Se il nodo viene distribuito in una sottorete privata, è in grado di comunicare con il cluster e altri servizi tramite un gateway NAT. AWS
> - Se le sottoreti non hanno accesso a Internet, leggi con attenzione le considerazioni e le aggiuntive descritte in Requisiti dei cluster privati.
> - Se si AWS Outposts selezionano le sottoreti Wavelength o Local Zone, le sottoreti non devono essere state passate al momento della creazione del cluster.

8. Confermare che la pila sia in grado di creare risorse IAM, quindi scegliere Crea pila.
9. Al termine della creazione della pila, selezionala nella console e scegli Outputs (Uscite).
10. Registra il NodeInstanceruolo per il gruppo di nodi che è stato creato. Ciò sarà utile quando configurerai i nodi Windows di Amazon EKS.

Fase 2: abilitazione dell'aggiunta di nodi al cluster

1. Verifica se disponi già di una `ConfigMap` per `aws-auth`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Se ti viene mostrata una `ConfigMap` per `aws-auth`, aggiornala se necessario.

   a. Apri `ConfigMap` per la modifica.

   ```
   kubectl edit -n kube-system configmap/aws-auth
   ```

   b. Aggiungi nuove voci `mapRoles`, se necessario. Imposta i `rolearn` valori sui valori del NodeInstanceruolo che hai registrato nelle procedure precedenti.

   ```
   [...]
   data:
     mapRoles: |
   - rolearn: <ARN of linux instance role (not instance profile)>
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
       - rolearn: <ARN of windows instance role (not instance profile)>
         username: system:node:{{EC2PrivateDNSName}}
         groups:
           - system:bootstrappers
           - system:nodes
           - eks:kube-proxy-windows
   [...]
   ```

   c. Salva il file ed esci dall'editor di testo.

3. Se hai ricevuto un messaggio di errore che indica "`Error from server (NotFound): configmaps "aws-auth" not found`", applica lo `ConfigMap` di stock.

   a. Scarica la mappa di configurazione.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/aws-auth-cm-windows.yaml
   ```

   b. Nel `aws-auth-cm-windows.yaml` file, impostate i `rolearn` valori sui valori NodeInstanceRole applicabili registrati nelle procedure precedenti. Per eseguire questa operazione, utilizza un editor di testo o sostituisci i *`example values`* tramite il comando seguente:

```
sed -i.bak -e 's|<ARN of linux instance role (not instance profile)>|my-node-linux-instance-role|' \
    -e 's|<ARN of windows instance role (not instance profile)>|my-node-windows-instance-role|' aws-auth-cm-windows.yaml
```

> **Important**
>
> - Non modificare altre righe in questo file.
> - Non utilizzare lo stesso ruolo IAM sia per i nodi Windows che per quelli Linux.

c. Applica la configurazione. L'esecuzione di questo comando potrebbe richiedere alcuni minuti.

```
kubectl apply -f aws-auth-cm-windows.yaml
```

4. Guarda lo stato dei nodi e attendi che raggiungano lo stato `Ready`.

```
kubectl get nodes --watch
```

Inserisci `Ctrl+C` per tornare a un prompt dello shell (interprete di comandi).

> **Note**
>
> Se ricevi qualsiasi altro errore di tipo di risorsa o autorizzazione, consulta la sezione Accesso negato o non autorizzato (`kubectl`) nell'argomento relativo alla risoluzione dei problemi.

Se i nodi di lavoro non riescono a unirsi al cluster, consulta Impossibile aggiungere i nodi al cluster nella Guida alla risoluzione dei problemi.

Fase 3: operazioni aggiuntive

1. (Facoltativo) Implementa un'applicazione di esempio per testare il cluster e i nodi Windows.
2. (Facoltativo) Se la policy IAM gestita AmazonEKS_CNI_Policy (se hai un cluster `IPv4`) o la policy *`AmazonEKS_CNI_IPv6_Policy`* (da te creata) se hai un cluster `IPv6`) è collegata

al the section called “Ruolo IAM del nodo”, ti consigliamo invece di assegnarla invece a un ruolo IAM associato all'account del servizio `aws-node` Kubernetes. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

3. Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

   - Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
   - No, Pods nel cluster è richiesto l'accesso al servizio di metadati dell'istanza Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

   Per ulteriori informazioni, consulta Limita l'accesso al profilo dell'istanza assegnato al nodo (worker).

Avvio dei nodi Ubuntu autogestiti

> **Note**
>
> I gruppi di nodi gestiti potrebbero offrire alcuni vantaggi per il tuo caso d'uso. Per ulteriori informazioni, consulta Gruppi di nodi gestiti.

Questo argomento descrive come avviare gruppi di Auto Scaling Ubuntusu nodi Amazon Elastic Kubernetes Service (EKS) o su nodi Ubuntu Pro Amazon Elastic Kubernetes Service (EKS) registrati nel cluster Amazon EKS. Ubuntue Ubuntu Pro per EKS si basano sul Ubuntu Minimal LTS ufficiale, includono il AWS kernel personalizzato sviluppato congiuntamente con EKS e sono stati creati appositamente per EKS. AWSUbuntu Proaggiunge una copertura di sicurezza aggiuntiva supportando i periodi di supporto estesi di EKS, il kernellivepatch, la conformità FIPS e la capacità di eseguire container illimitati. Pro

Dopo che i nodi si sono uniti al cluster, è possibile distribuirvi applicazioni containerizzate. Per ulteriori informazioni, consulta la documentazione relativa al supporto per le AMI Ubuntuaccese AWS e personalizzate contenuta nella `eksctl` documentazione.

Important

- I nodi Amazon EKS sono istanze Amazon EC2 standard e la loro fatturazione è basata sui normali prezzi dell'istanza Amazon EC2. Per ulteriori informazioni, consulta Prezzi di Amazon EC2.
- Puoi avviare Ubuntu nodi nei cluster estesi di Amazon EKS su AWS Outposts, ma non puoi avviarli in cluster AWS locali su Outposts. Per ulteriori informazioni, consulta Amazon EKS su AWS Outposts.
- È possibile eseguire l'implementazione su istanze Amazon EC2 con processori x86 o Arm. Tuttavia, le istanze dotate di Inferentia chip potrebbero dover prima installare l'SDK. Neuron

Da avviare Ubuntu per EKS o Ubuntu Pro per nodi EKS utilizzando **`eksctl`**

Questa procedura richiede `eksctl` versione `0.183.0` o successiva. Puoi verificare la versione con il comando seguente:

```
eksctl version
```

Per istruzioni sull'installazione o sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Note

Questa procedura funziona solo per i cluster creati con `eksctl`.

1. Copia i seguenti contenuti sul dispositivo. Sostituisci `my-cluster` con il nome del cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfabetico e non può avere una lunghezza superiore a 100 caratteri. Sostituisci `ng-ubuntu` con un nome per il gruppo di nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura. Per eseguire la distribuzione sulle Arm istanze, sostituiscila `m5.large` con un tipo di Arm istanza. Sostituisci `my-ec2-keypair-name` con il nome di una coppia di chiavi SSH Amazon EC2; che puoi utilizzare per connetterti utilizzando SSH nei nodi dopo l'avvio. Se non disponi di una coppia di chiavi Amazon EC2,

puoi crearla nella AWS Management Console. Per ulteriori informazioni, consulta le coppie di chiavi Amazon EC2 nella Guida per l'utente di Amazon EC2. Sostituisci tutti i *example values* rimanenti con i valori in tuo possesso. Dopo aver effettuato le sostituzioni, eseguire il comando modificato per creare `ubuntu.yaml`.

> **Important**
>
> Per distribuire un gruppo di nodi nelle sottoreti della zona AWS locale AWS Outposts AWS Wavelength, don't pass AWS Outposts AWS Wavelength, o nelle sottoreti della zona AWS locale al momento della creazione del cluster. È necessario specificare le sottoreti nell'esempio seguente. Per ulteriori informazioni, vedere Creare un gruppo di nodi da un file di configurazione e lo Schema del file config nella documentazione su `eksctl`. Sostituiscilo *region-code* con quello in cui si trova il Regione AWS cluster.

```
cat >ubuntu.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: '1.30'

iam:
  withOIDC: true

nodeGroups:
  - name: ng-ubuntu
    instanceType: m5.large
    desiredCapacity: 3
    amiFamily: Ubuntu22.04
    ami: auto-ssm
    iam:
       attachPolicyARNs:
          - arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
          - arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
          - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
          - arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
    ssh:
```

```
        allow: true
        publicKeyName: my-ec2-keypair-name
EOF
```

Per creare un gruppo di Ubuntu Pro nodi, è sufficiente modificare il `amiFamily` valore in`UbuntuPro2204`.

2. Implementare i nodi mediante il comando seguente.

```
eksctl create nodegroup --config-file=ubuntu.yaml
```

Di seguito viene riportato un output di esempio:

Durante la creazione dei nodi vengono generate diverse righe. Una delle ultime righe di output è simile alla seguente riga di esempio.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facoltativo) implementa un'applicazione di esempio per testare i nodi Ubuntu.
4. Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

   - Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
   - Nessuno Pods nel cluster richiede l'accesso al servizio di metadati delle istanze Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

   Per ulteriori informazioni, consulta Limita l'accesso al profilo dell'istanza assegnato al nodo (worker).

Aggiornamenti del nodo autogestito

Quando viene rilasciata una nuova AMI ottimizzata per Amazon EKS, considerare la sostituzione dei nodi nel gruppo di nodi autogestiti con la nuova AMI. Analogamente, se hai aggiornato la versione Kubernetes per il cluster Amazon EKS, aggiorna i nodi per utilizzarli con la stessa versione Kubernetes.

> **Important**
>
> In questo argomento vengono descritti gli aggiornamenti dei nodi per i gruppi di nodi autogestiti. Se stai usando Gruppi di nodi gestiti, vedi Aggiornamento di un gruppo di nodi gestiti.

Esistono due modi di base per aggiornare i gruppi di nodi autogestiti nei cluster per utilizzare una nuova AMI:

Migrazione a un nuovo gruppo di nodi

Creare un nuovo gruppo di nodi ed eseguire la migrazione dei Pods a quel gruppo. La migrazione a un nuovo gruppo di nodi è più aggraziata del semplice aggiornamento dell'ID AMI in una pila AWS CloudFormation esistente. Questo perché il processo di migrazione esclude il vecchio gruppo di nodi come `NoSchedule` e svuota i nodi dopo che un nuovo stack è pronto ad accettare il carico di lavoro dei Pod esistente.

Aggiornamento di un gruppo di nodi autogestiti esistente

Aggiornare lo stack AWS CloudFormation per un gruppo di nodi esistente di modo che utilizzi la nuova AMI. Questo metodo non è supportato per i gruppi di nodi creati con `eksctl`.

Migrazione a un nuovo gruppo di nodi

Questo argomento consente di creare un nuovo gruppo di nodi, di migrare correttamente le applicazioni esistenti al nuovo gruppo e di rimuovere il precedente gruppo di nodi dal cluster. È possibile eseguire la migrazione a un nuovo gruppo di nodi utilizzando `eksctl` o la AWS Management Console.

eksctl

Migrazione delle applicazioni verso un nuovo gruppo di nodi con **`eksctl`**

Per ulteriori informazioni sull'utilizzo di eksctl per la migrazione, vedere Unmanaged nodegroups nella documentazione. `eksctl`

Questa procedura richiede `eksctl` versione `0.183.0` o successiva. Puoi verificare la versione con il comando seguente:

```
eksctl version
```

Per istruzioni sull'installazione o sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

> **Note**
>
> Questa procedura funziona solo per i cluster e per i gruppi di nodi creati con `eksctl`.

1. Recuperare il nome dei gruppi di nodi esistenti, sostituendo *`my-cluster`* con il nome del cluster.

   ```
   eksctl get nodegroups --cluster=my-cluster
   ```

   Di seguito viene riportato un output di esempio:

   ```
   CLUSTER      NODEGROUP          CREATED               MIN SIZE      MAX SIZE
    DESIRED CAPACITY     INSTANCE TYPE     IMAGE ID
   default      standard-nodes   2019-05-01T22:26:58Z  1             4             3
                       t3.medium         ami-05a71d034119ffc12
   ```

2. Avvia un nuovo gruppo di nodi con `eksctl` con il comando seguente. Nel comando, sostituire ogni *`example value`* con i propri valori. Il numero della versione non può essere successivo alla versione di Kubernetes per il piano di controllo (control-plane). Inoltre, non può essere più di due versioni secondarie precedenti rispetto alla versione Kubernetes per il piano di controllo (control-plane). Si consiglia di utilizzare la stessa versione del piano di controllo.

   Consigliamo di bloccare l'accesso dei Pod a IMDS se si verificano le seguenti condizioni:

   - Prevedi di assegnare ruoli IAM a tutti gli account di servizio Kubernetes in modo che i Pods dispongano solo delle autorizzazioni minime necessarie.
   - No, Pods nel cluster è richiesto l'accesso al servizio di metadati dell'istanza Amazon EC2 (IMDS) per altri motivi, come il recupero della corrente. Regione AWS

   Per ulteriori informazioni, consulta Limitazione dell'accesso al profilo dell'istanza assegnato al nodo worker.

Per bloccare l'accesso del Pod a IMDS, aggiungi l'opzione `--disable-pod-imds` al seguente comando.

> **Note**
>
> Per ulteriori flag disponibili e relative descrizioni, vedere https://eksctl.io/.

```
eksctl create nodegroup \
  --cluster my-cluster \
  --version 1.30 \
  --name standard-nodes-new \
  --node-type t3.medium \
  --nodes 3 \
  --nodes-min 1 \
  --nodes-max 4 \
  --managed=false
```

3. Quando il comando precedente viene completato, verificare che tutti i nodi abbiano raggiunto lo stato `Ready` con il comando seguente:

```
kubectl get nodes
```

4. Eliminare il gruppo di nodi originale con il comando seguente. Nel comando, sostituire ogni *`example value`* con i nomi dei cluster e dei gruppi di nodi:

```
eksctl delete nodegroup --cluster my-cluster --name standard-nodes-old
```

AWS Management Console and AWS CLI

Per migrare le applicazioni in un nuovo gruppo di nodi con e AWS Management ConsoleAWS CLI

1. Avviare un nuovo gruppo di nodi seguendo i passaggi descritti in Avvio di nodi Amazon Linux autogestiti.
2. Al termine della creazione della pila, selezionala nella console e scegli Outputs (Uscite).
3. Registra il NodeInstanceruolo per il gruppo di nodi che è stato creato. Ciò è necessario per aggiungere i nuovi nodi Amazon EKS al cluster.

> **Note**
>
> Se si dispone di policy IAM aggiuntive associate al vecchio ruolo IAM del gruppo di nodi, associa le stesse policy al nuovo ruolo IAM del gruppo di nodi per mantenere tale funzionalità nel nuovo gruppo. Questo vale, ad esempio, se hai aggiunto le autorizzazioni per il Cluster Autoscaler di Kubernetes.

4. Aggiorna i gruppi di sicurezza per entrambi i gruppi di nodi in modo che possano comunicare tra loro. Per ulteriori informazioni, consulta Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS.

   a. Registra gli ID del gruppo di sicurezza per entrambi i gruppi dei nodi. Questo viene mostrato come valore del NodeSecuritygruppo negli output dello AWS CloudFormation stack.

      È possibile utilizzare i seguenti AWS CLI comandi per ottenere gli ID dei gruppi di sicurezza dai nomi dello stack. In questi comandi, `oldNodes` è il nome AWS CloudFormation dello stack per lo stack di nodi precedente ed `newNodes` è il nome dello stack verso cui state migrando. Sostituisci ogni *`example value`* con i valori in tuo possesso.

      ```
      oldNodes="old_node_CFN_stack_name"
      newNodes="new_node_CFN_stack_name"

      oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name
       $oldNodes \
      --query 'StackResources[?
      ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
      --output text)
      newSecGroup=$(aws cloudformation describe-stack-resources --stack-name
       $newNodes \
      --query 'StackResources[?
      ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
      --output text)
      ```

   b. Aggiungi regole in ingresso per ciascun gruppo di sicurezza del nodo in modo da accettare il traffico tra loro.

I AWS CLI comandi seguenti aggiungono regole in entrata a ciascun gruppo di sicurezza che consentono tutto il traffico su tutti i protocolli dell'altro gruppo di sicurezza. Ciò consente ai Pods in ogni gruppo di nodi di comunicare tra loro durante la migrazione del carico di lavoro verso il nuovo gruppo.

```
aws ec2 authorize-security-group-ingress --group-id $oldSecGroup \
--source-group $newSecGroup --protocol -1
aws ec2 authorize-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

5. Modifica la configmap `aws-auth` per mappare il nuovo ruolo dell'istanza del nodo in RBAC.

```
kubectl edit configmap -n kube-system aws-auth
```

Aggiungi una nuova voce `mapRoles` per il nuovo gruppo di nodi. Se il cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

```
apiVersion: v1
data:
  mapRoles: |
    - rolearn: ARN of instance role (not instance profile)
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes>
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-16-NodeInstanceRole-
U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
```

Sostituisci lo *ARN of instance role (not instance profile)* snippet con il valore del NodeInstanceruolo che hai registrato in un passaggio precedente. Quindi, salva e chiudi il file per applicare la configmap aggiornata.

6. Guarda lo stato dei nodi e attendi fino a quando i nuovi nodi si uniscono al cluster e raggiungono lo stato `Ready`.

```
kubectl get nodes --watch
```

7. (Facoltativo) Se si sta utilizzando il Cluster Autoscaler di Kubernetes, ridurre l'implementazione fino a raggiungere zero (0) repliche per evitare azioni di dimensionamento conflittuali.

```
kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
```

8. Utilizzare il comando seguente per il taint di ciascuno dei nodi che si desidera rimuovere con `NoSchedule`. In questo modo i nuovi Pods non vengono pianificati o riprogrammati sui nodi che stai sostituendo. Per ulteriori informazioni consulta Taint e tolleranze nella documentazione di Kubernetes.

```
kubectl taint nodes node_name key=value:NoSchedule
```

   Se si aggiornano i nodi a una nuova versione di Kubernetes, è possibile identificare e fare il taint di tutti i nodi di una determinata versione di Kubernetes (in questo caso, `1.28`) con il frammento di codice seguente. Il numero della versione non può essere successivo alla versione di Kubernetes per il piano di controllo (control-plane). Inoltre, non può essere più di due versioni secondarie precedenti rispetto alla versione Kubernetes del piano di controllo (control-plane). Si consiglia di utilizzare la stessa versione del piano di controllo.

```
K8S_VERSION=1.28
nodes=$(kubectl get nodes -o jsonpath="{.items[?
(@.status.nodeInfo.kubeletVersion==\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Tainting $node"
    kubectl taint nodes $node key=value:NoSchedule
done
```

9. Determinare il provider DNS del cluster.

```
kubectl get deployments -l k8s-app=kube-dns -n kube-system
```

   Di seguito viene riportato un output di esempio: Questo cluster utilizza CoreDNS per la risoluzione DNS, ma il cluster può invece restituire `kube-dns`):

```
NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
coredns   1         1         1            1           31m
```

10. Se l'implementazione corrente è in esecuzione per un numero di volte inferiore a 2 repliche, scalare la distribuzione a 2 repliche. Sostituire *coredns* con **kubedns** se l'output del comando precedente ha avuto tale risultato.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

11. Svuotare ciascuno dei nodi che si desidera rimuovere dal cluster con il comando seguente:

```
kubectl drain node_name --ignore-daemonsets --delete-local-data
```

Se si aggiornano i nodi a una nuova versione di Kubernetes, identificare e svuotare tutti i nodi di una determinata versione di Kubernetes (in questo caso, *1.28*) con il frammento di codice seguente.

```
K8S_VERSION=1.28
nodes=$(kubectl get nodes -o jsonpath="{.items[?
(@.status.nodeInfo.kubeletVersion==\"v$K8S_VERSION\")].metadata.name}")
for node in ${nodes[@]}
do
    echo "Draining $node"
    kubectl drain $node --ignore-daemonsets --delete-local-data
done
```

12. Una volta terminata tale operazione, revocare le regole in ingresso del gruppo di sicurezza autorizzate in precedenza. Quindi, elimina lo AWS CloudFormation stack per terminare le istanze.

> **Note**
>
> Se hai collegato delle policy IAM aggiuntive al tuo vecchio gruppo di nodi (ruolo IAM, ad esempio aggiungendo autorizzazioni per Kubernetes Cluster Autoscaler), scollega quelle policy aggiuntive dal ruolo prima di poter eliminare lo stack. AWS CloudFormation

a. Revocare le regole in ingresso create in precedenza per i gruppi di sicurezza dei nodi. In questi comandi, `oldNodes` c'è il nome AWS CloudFormation dello stack di nodi precedente ed `newNodes` è il nome dello stack verso cui stai migrando.

```
oldNodes="old_node_CFN_stack_name"
newNodes="new_node_CFN_stack_name"

oldSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $oldNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
newSecGroup=$(aws cloudformation describe-stack-resources --stack-name
 $newNodes \
--query 'StackResources[?
ResourceType==`AWS::EC2::SecurityGroup`].PhysicalResourceId' \
--output text)
aws ec2 revoke-security-group-ingress --group-id $oldSecGroup \
--source-group $newSecGroup --protocol -1
aws ec2 revoke-security-group-ingress --group-id $newSecGroup \
--source-group $oldSecGroup --protocol -1
```

b. Apri la console all'indirizzo https://console.aws.amazon.com/cloudformation AWS CloudFormation .

c. Selezionare la pila del nodo precedente.

d. Scegli Elimina.

e. Nella finestra di dialogo di conferma Delete stack (Elimina stack) scegliere Delete stack. (Elimina stack).

13. Modifica il configmap `aws-auth` per rimuovere il precedente ruolo dell'istanza del nodo da RBAC.

```
kubectl edit configmap -n kube-system aws-auth
```

Eliminare la voce `mapRoles` per il precedente gruppo di nodo. Se il cluster si trova negli AWS GovCloud Stati Uniti orientali o AWS GovCloud negli Stati Uniti occidentali Regioni AWS, `arn:aws:` sostituiscilo con. `arn:aws-us-gov:`

```
apiVersion: v1
```

```
data:
  mapRoles: |
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-16-NodeInstanceRole-
W70725MZQFF8
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
    - rolearn: arn:aws:iam::111122223333:role/nodes-1-15-NodeInstanceRole-
U11V27W93CX5
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes>
```

Salvare e chiudere il file per applicare la configmap aggiornata.

14. (Facoltativo) Se si sta utilizzando il Cluster Autoscaler di Kubernetes, dimensionare l'implementazione a 1 replica.

> **Note**
>
> È inoltre necessario applicare i tag al nuovo gruppo Auto Scaling in modo appropriato (ad esempio, `k8s.io/cluster-autoscaler/enabled,k8s.io/cluster-autoscaler/`*`my-cluster`*) e aggiornare il comando di implementazione di Cluster Autoscaler per puntare al nuovo gruppo Auto Scaling contrassegnato. Per ulteriori informazioni, consulta Cluster Autoscaler on. AWS

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

15. (Facoltativo) Verificare che si sta utilizzando la versione più recente del plug-in CNI di Amazon VPC per Kubernetes. Potrebbe essere necessario aggiornare la versione CNI per utilizzare i tipi di istanze supportati più recenti. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.
16. Se il cluster utilizza `kube-dns` per la risoluzione DNS (vedere il passaggio precedente), ridurre orizzontalmente l'implementazione `kube-dns` a una replica.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

Aggiornamento di un gruppo di nodi autogestiti esistente

Questo argomento descrive come aggiornare uno stack di nodi AWS CloudFormation autogestito esistente con una nuova AMI. È possibile utilizzare questa procedura per aggiornare i nodi a una nuova versione di Kubernetes in seguito all'aggiornamento di un cluster. In caso contrario, è possibile eseguire l'aggiornamento all'AMI ottimizzata per Amazon EKS più recente per una versione Kubernetes esistente.

> **Important**
>
> In questo argomento vengono descritti gli aggiornamenti dei nodi per i gruppi di nodi autogestiti. Per ulteriori informazioni sull'uso di Gruppi di nodi gestiti, consultare Aggiornamento di un gruppo di nodi gestiti.

L'ultimo AWS CloudFormation modello di nodo Amazon EKS predefinito è configurato per avviare un'istanza con la nuova AMI nel cluster prima di rimuoverne una vecchia, una alla volta. Questa configurazione garantisce sempre il conteggio desiderato del gruppo Auto Scaling delle istanze attive nel cluster durante l'aggiornamento in sequenza.

> **Note**
>
> Questo metodo non è supportato per i gruppi di nodi creati con `eksctl`. Se è stato creato il cluster o il gruppo di nodi con `eksctl`, consultare Migrazione a un nuovo gruppo di nodi.

Per aggiornare un gruppo di nodi esistente

1. Determinare il provider DNS del cluster.

   ```
   kubectl get deployments -l k8s-app=kube-dns -n kube-system
   ```

   Di seguito viene riportato un output di esempio: Questo cluster utilizza CoreDNS per la risoluzione DNS, ma il cluster può invece restituire `kube-dns`. L'output potrebbe avere un aspetto diverso a seconda della versione di `kubectl` che stai utilizzando.

   ```
   NAME      DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
   coredns   1         1         1            1           31m
   ```

2. Se l'implementazione corrente è in esecuzione per un numero di volte inferiore a 2 repliche, scalare la distribuzione a 2 repliche. Sostituire *coredns* con **kube-dns** se l'output del comando precedente ha avuto tale risultato.

```
kubectl scale deployments/coredns --replicas=2 -n kube-system
```

3. (Facoltativo) Se si sta utilizzando il Cluster Autoscaler di Kubernetes, ridurre l'implementazione fino a raggiungere zero (0) repliche per evitare azioni di dimensionamento conflittuali.

```
kubectl scale deployments/cluster-autoscaler --replicas=0 -n kube-system
```

4. Stabilire il tipo di istanza e il conteggio di istanze desiderato dell'attuale gruppo di nodi. Inserisci questi valori in un secondo momento, quando aggiorni il AWS CloudFormation modello per il gruppo.

   a. Apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.

   b. Nel pannello di navigazione a sinistra, scegli Launch Configurations (Configurazioni di avvio) e prendi nota del tipo di istanza per la configurazione di avvio del nodo esistente.

   c. Nel pannello di navigazione a sinistra, scegli Auto Scaling Groups (Gruppi Auto Scaling) e prendi nota del conteggio delle istanze Desired (Desiderato) per il gruppo Auto Scaling del nodo.

5. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.

6. Selezionare la pila del gruppo di nodi, quindi scegliere Aggiorna.

7. Selezionare Replace current template (Sostituisci modello corrente) e scegliere Amazon S3 URL (URL Amazon S3).

8. Per l'URL di Amazon S3, incolla il seguente URL nell'area di testo per assicurarti di utilizzare la versione più recente del modello di nodo AWS CloudFormation . Quindi scegliere Next (Successivo):

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/amazon-eks-nodegroup.yaml
```

9. Nella pagina Specify stack details (Specifica dettagli pila), compilare i parametri seguenti e scegliere Next (Successivo):

- NodeAutoScalingGroupDesiredCapacity— Inserisci il numero di istanze desiderato che hai registrato in un passaggio precedente. In alternativa, inserire il nuovo numero desiderato di nodi come riferimento del dimensionamento quando viene aggiornata la pila.
- NodeAutoScalingGroupMaxSize— Inserisci il numero massimo di nodi a cui il gruppo Auto Scaling del nodo può scalare orizzontalmente. Questo valore deve essere almeno un nodo in più rispetto alla capacità desiderata. Ciò consente di eseguire un aggiornamento in sequenza dei nodi senza ridurre il numero di nodi durante l'aggiornamento.
- NodeInstanceTipo: scegli il tipo di istanza registrato nel passaggio precedente. In alternativa, scegliere un tipo di istanza diverso per i nodi. Prima di scegliere un tipo di istanza diverso, rivedere Scelta di un tipo di istanza Amazon EC2. Ogni tipo di istanza Amazon EC2 supporta un numero massimo di interfacce di rete elastiche (interfaccia di rete) e ogni interfaccia di rete supporta un numero massimo di indirizzi IP. Poiché a ogni nodo worker e Pod viene assegnato il proprio indirizzo IP, è importante scegliere un tipo di istanza che supporti il numero massimo di Pods che si desidera eseguire su ciascun nodo Amazon EC2. Per un elenco del numero di interfacce di rete e di indirizzi IP supportati dai tipi di istanza, consulta Indirizzi IP per interfaccia di rete e per tipo di istanza. Ad esempio, il tipo di istanza `m5.large` supporta un massimo di 30 indirizzi IP per il nodo worker e i Pods.

  > **Note**
  >
  > I tipi di istanze supportati per l'ultima versione del Amazon VPC CNI plugin for Kubernetes sono visualizzati in vpc_ip_resource_limit.go su GitHub. Potrebbe essere necessario aggiornare la versione Amazon VPC CNI plugin for Kubernetes per utilizzare i tipi di istanze supportati più recenti. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.

  > **Important**
  >
  > Alcuni tipi di istanza potrebbero non essere disponibili in tutti Regioni AWS.

- NodeImageIDSSMParam — Il parametro Amazon EC2 Systems Manager dell'ID AMI a cui desideri eseguire l'aggiornamento. Il valore seguente utilizza l'AMI ottimizzata per Amazon EKS più recente per Kubernetes versione `1.30`.

```
/aws/service/eks/optimized-ami/1.30/amazon-linux-2/recommended/image_id
```

È possibile sostituire *1.30* con una stessa versione Kubernetes supportata. In alternativa, potrebbe essere fino a una versione precedente alla versione di Kubernetes in esecuzione sul piano di controllo (control-plane). Si consiglia di mantenere i nodi alla stessa versione del piano di controllo. Puoi anche sostituirlo *amazon-linux-2* con un altro tipo di AMI. Per ulteriori informazioni, consulta Recupero ID delle AMI Amazon Linux ottimizzate per Amazon EKS.

> **Note**
>
> L'utilizzo del parametro Amazon EC2 Systems Manager consente di aggiornare i nodi in futuro senza dover cercare e specificare un ID AMI. Se lo stack AWS CloudFormation sta utilizzando questo valore, qualsiasi aggiornamento dello stack avvia sempre l'AMI ottimizzata per Amazon EKS consigliata più recente per la versione Kubernetes specificata. Questo è il caso anche se non si modificano valori nel modello.

- NodeImageID: per utilizzare un'AMI personalizzata, inserisci l'ID dell'AMI da utilizzare.

> **Important**
>
> Questo valore sostituisce qualsiasi valore specificato per NodeImage IDSSMParam. Se desideri utilizzare il valore NodeImageIDSSMParam, assicurati che il valore di Id sia vuoto. NodeImage

- DisabilitaIMDSv1: ogni nodo supporta Instance Metadata Service versione 1 (IMDSv1) e IMDSv2 per impostazione predefinita. Tuttavia, è possibile disabilitare IMDSv1. Seleziona true se non desideri che alcun nodo o Pods pianificato sul gruppo di nodi utilizzi IMDSv1. Per ulteriori informazioni su IMDS, consulta Configurazione del servizio di metadati dell'istanza. Se hai implementato i ruoli IAM per gli account di servizio, assegna le autorizzazioni necessarie direttamente a tutti Pods coloro che richiedono l'accesso ai servizi. AWS In questo modo, nessun Pods membro del cluster richiederà l'accesso a IMDS per altri motivi, come il recupero della corrente. Regione AWS Quindi, è anche possibile disabilitare l'accesso a IMDSv2 per i Pods che non utilizzano la rete host. Per ulteriori informazioni, consulta Limita l'accesso al profilo di istanza assegnato al nodo (worker).

10. (Facoltativo) Nella pagina Options (Opzioni), contrassegna con dei tag le risorse della pila. Seleziona Successivo.
11. Nella pagina Verifica, esaminare le informazioni, confermare che la pila è in grado di creare risorse IAM, quindi scegliere Aggiorna pila.

> **Note**
>
> L'aggiornamento di ogni nodo nel cluster richiede diversi minuti. Attendi il completamento dell'aggiornamento di tutti i nodi prima di eseguire la procedura successiva.

12. Se il provider DNS del cluster è `kube-dns`, dimensionare l'implementazione di `kube-dns` a 1 replica.

```
kubectl scale deployments/kube-dns --replicas=1 -n kube-system
```

13. (Facoltativo) Se si sta utilizzando il Cluster Autoscaler di Kubernetes, dimensionare l'implementazione al numero di repliche desiderato.

```
kubectl scale deployments/cluster-autoscaler --replicas=1 -n kube-system
```

14. (Facoltativo) Accertati di utilizzare la versione più recente del Amazon VPC CNI plugin for Kubernetes. Potrebbe essere necessario aggiornare la versione del Amazon VPC CNI plugin for Kubernetes per utilizzare i tipi di istanze supportati più recenti. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.

AWS Fargate

> **Important**
>
> AWS Fargate con Amazon EKS non è disponibile negli AWS GovCloud Stati Uniti orientali e AWS GovCloud negli Stati Uniti occidentali.

In questa sezione viene illustrato l'uso di Amazon EKS per eseguire i Pods Kubernetes su AWS Fargate. Fargate è una tecnologia che fornisce capacità di calcolo on demand e di dimensioni adeguate per i container. Con Fargate, non è più necessario effettuare personalmente il provisioning, la configurazione o il dimensionamento dei gruppi di macchine virtuali per eseguire i container. Viene

anche eliminata la necessità di scegliere i tipi di server, di decidere quando dimensionare i gruppi di nodi o di ottimizzare l'impacchettamento dei cluster.

Puoi controllare quali Pods sono avviati su Fargate e come vengono eseguiti con i profili Fargate. I profili Fargate vengono definiti nell'ambito del cluster Amazon EKS. Amazon EKS si integra Kubernetes con Fargate utilizzando controller creati utilizzando il modello upstream ed estensibile fornito AWS da. Kubernetes Questi controller vengono eseguiti nell'ambito del piano di controllo Kubernetes gestito da Amazon EKS e sono responsabili della pianificazione dei Pods Kubernetes nativi su Fargate. I controller Fargate includono un nuovo pianificatore che viene eseguito insieme al pianificatore predefinito di Kubernetes, oltre a diversi controller di ammissione mutanti e convalidanti. Quando si avvia un Pod che soddisfa i criteri per l'esecuzione su Fargate, i controller Fargate in esecuzione nel cluster riconoscono, aggiornano e pianificano il Pod su Fargate.

In questa sezione sono descritti i diversi componenti dei Pods in esecuzione su Fargate e sono riportate considerazioni speciali per l'utilizzo di Fargate con Amazon EKS.

AWS Fargate considerazioni

Di seguito sono elencati alcuni punti da considerare sull'uso di Fargate in Amazon EKS.

- Ogni Pod in esecuzione su Fargate ha un proprio limite di isolamento. Non condividono il kernel sottostante, le risorse CPU, le risorse di memoria o l'interfaccia di rete elastica con un altro Pod.
- I Network Load Balancer e gli Application Load Balancer (ALB) possono essere utilizzati solo con Fargate con destinazioni IP. Per ulteriori informazioni, consultare Creazione di un Network Load Balancer e Bilanciamento del carico di applicazione su Amazon EKS.
- I servizi esposti Fargate vengono eseguiti solo in modalità IP di tipo destinazione e non in modalità IP del nodo. Il modo consigliato per verificare la connettività da un servizio in esecuzione su un nodo gestito e da un servizio in esecuzione su Fargate è quello di connettersi tramite il nome del servizio.
- Nel momento in cui sono programmati per l'esecuzione su Fargate, i pod devono corrispondere a un profilo Fargate. I pod che non corrispondono a un profilo Fargate potrebbero rimanere bloccati nello stato `Pending`. Se esiste un profilo Fargate corrispondente, puoi eliminare i Pods in sospeso creati per riprogrammarli su Fargate.
- I Daemonset non sono supportati su Fargate. Se l'applicazione richiede un daemon, bisogna riconfigurare quel daemon per essere eseguito come container sidecar nei Pods .
- I container privilegiati non sono supportati su Fargate.

- I pod in esecuzione su Fargate non possono specificare `HostPort` o `HostNetwork` nel manifesto Pod.
- Per i Pods Fargate, Il limite flessibile predefinito di `nofile` e `nproc` è 1024 mentre il limite rigido è 65535.
- Le GPU non sono attualmente disponibili su Fargate.
- I pod che funzionano su Fargate sono supportati solo su sottoreti private (con accesso AWS gateway NAT ai servizi, ma non un percorso diretto verso un Internet Gateway), quindi il VPC del cluster deve disporre di sottoreti private. Per i cluster senza accesso Internet in uscita, consulta Requisiti dei cluster privati.
- Puoi utilizzare Vertical Pod Autoscaler per impostare la dimensione corretta iniziale della CPU e la memoria per i Pods Fargate, quindi utilizzare il comando Horizontal Pod Autoscaler per scalare quei Pods. Se desideri che il Vertical Pod Autoscaler implementi nuovamente in automatico i Pods su Fargate con combinazioni di CPU e memoria maggiori, imposta la modalità per il Vertical Pod Autoscaler su `Auto` o `Recreate` per garantire il corretto funzionamento. Per ulteriori informazioni, consulta la documentazione di Vertical Pod Autoscaler su GitHub.
- La risoluzione DNS e i nomi host DNS devono essere abilitati per il VPC. Per ulteriori informazioni, consulta Visualizzazione e aggiornamento del supporto DNS per il VPC.
- Amazon EKS Fargate aggiunge defense-in-depth Kubernetes applicazioni isolando ogni Pod all'interno di una macchina virtuale (VM). Questo limite VM impedisce l'accesso alle risorse basate su host utilizzate da altri pod in caso di evasione da un container, un metodo comune per attaccare le applicazioni containerizzate e ottenere l'accesso a risorse esterne al container.

  L'utilizzo di Amazon EKS non modifica le tue responsabilità ai sensi del modello di responsabilità condivisa. È necessario considerare attentamente la configurazione dei controlli di sicurezza e gestione del cluster. Il modo più sicuro per isolare un'applicazione è sempre quello di eseguirla in un cluster separato.
- I profili Fargate supportano la specificazione di sottoreti da blocchi CIDR secondari del VPC. È possibile specificare un blocco CIDR secondario. Ciò è necessario perché in una sottorete è disponibile un numero limitato di indirizzi IP. Di conseguenza, nel cluster è possibile creare un numero limitato di Pods. L'uso di sottoreti diverse per i Pods consente di aumentare il numero di indirizzi IP disponibili. Per ulteriori informazioni, consulta Aggiunta di blocchi CIDR `IPv4` a un VP.
- Il servizio metadati dell'istanza Amazon EC2 (IMDS) non è disponibile per i Pods implementati sui nodi Fargate. Se disponi di Pods distribuiti in Fargate che necessitano di credenziali IAM, assegnale ai Podsutilizzando Ruoli IAM per gli account di servizio. Se i Pods hanno bisogno di accedere ad altre informazioni disponibili tramite IMDS, bisogna eseguire la codifica fissa di queste

informazioni nelle specifiche del Pod. Ciò include la Regione AWS o la zona di disponibilità in cui Pod viene distribuito a.

- Non puoi distribuire Pods Fargate su AWS Wavelength, o AWS Outposts Local AWS Zones.
- Periodicamente, Amazon EKS deve applicare patch ai Pods per garantire che siano sicuri. Gli aggiornamenti vengono effettuati in modo da ridurre l'impatto, ma in alcuni casi i Pods devono essere eliminati se non vengono espulsi correttamente. Di seguito sono riportate le azioni che è possibile intraprendere per ridurre al minimo le interruzioni. Per ulteriori informazioni, consulta Applicazione di patch del sistema operativo Fargate.
- Il plug-in CNI di Amazon VPC PER Amazon EKS è installato sui nodi Fargate. Non puoi utilizzare Plugin CNI compatibili alternativi con nodi Fargate.
- Un Pod in esecuzione su Fargate monta automaticamente un file system Amazon EFS. Non è possibile utilizzare il provisioning dinamico dei volumi persistenti con nodi Fargate, ma è possibile utilizzare il provisioning statico.
- Non puoi montare volumi Amazon EBS su Fargate Pods.
- Puoi eseguire il controller Amazon EBS CSI sui nodi Fargate, ma sul nodo Amazon EBS CSI DaemonSet può essere eseguito solo su istanze Amazon EC2.
- Dopo che un Job Kubernetes è stato contrassegnato `Completed` o `Failed`, i Pods creati dal Job normalmente continuano a esistere. Questo comportamento ti consente di visualizzare i tuoi log e risultati, ma con Fargate dovrai sostenere dei costi se non ripulisci il Job in seguito.

  Per eliminare automaticamente i dati correlati Pods dopo un Job completamento o un errore, puoi specificare un periodo di tempo utilizzando il controller time-to-live (TTL). L'esempio seguente mostra la specifica di `.spec.ttlSecondsAfterFinished` nel manifesto del Job.

```
apiVersion: batch/v1
kind: Job
metadata:
  name: busybox
spec:
  template:
    spec:
      containers:
      - name: busybox
        image: busybox
        command: ["/bin/sh", "-c", "sleep 10"]
      restartPolicy: Never
  ttlSecondsAfterFinished: 60 # <-- TTL controller
```

Guida introduttiva all' AWS Fargate uso di Amazon EKS

> **Important**
>
> AWS Fargate con Amazon EKS non è disponibile negli AWS GovCloud Stati Uniti orientali e AWS GovCloud negli Stati Uniti occidentali.

Questo argomento descrive come iniziare a utilizzare il tuo cluster Amazon EKS. Pods AWS Fargate

Se limiti l'accesso all'endpoint pubblico del cluster utilizzando blocchi CIDR, consigliamo di abilitare anche l'accesso agli endpoint privati. In questo modo Fargate Pods può comunicare con il cluster. Senza l'endpoint privato abilitato, i blocchi CIDR specificati per l'accesso pubblico devono includere le origini di uscita dal VPC. Per ulteriori informazioni, consulta Controllo accessi all'endpoint del cluster Amazon EKS.

Prerequisito

Un cluster esistente. Se non disponi già di un cluster Amazon EKS, consulta Guida introduttiva ad Amazon EKS.

Verifica che i nodi esistenti possano comunicare con i Pods Fargate

Se utilizzi un nuovo cluster senza nodi di lavoro o un cluster con solo gruppi di nodi gestiti, puoi passare a Creazione di un ruolo di esecuzione del Pod Fargate.

Poniamo che tu stia lavorando con un cluster esistente che dispone già di nodi associati. Assicurati che Pods su questi nodi possa comunicare liberamente con il Pods in esecuzione su Fargate. I Pods che sono in esecuzione su Fargate vengono configurati automaticamente per utilizzare il gruppo di sicurezza del cluster a cui sono associati. Devi verificare che tutti i nodi esistenti nel cluster possano inviare e ricevere traffico da e verso il gruppo di sicurezza del cluster. I Gruppi di nodi gestiti vengono configurati in automatico per utilizzare anche il gruppo di sicurezza del cluster, pertanto non è necessario modificarli o verificarli per garantire tale compatibilità.

Per i gruppi di nodi esistenti che sono stati creati con `eksctl` o i AWS CloudFormation modelli gestiti di Amazon EKS, puoi aggiungere manualmente il gruppo di sicurezza del cluster ai nodi. In alternativa, è possibile modificare il modello di avvio del gruppo Auto Scaling per il gruppo di nodi per collegare il gruppo di sicurezza cluster alle istanze. Per ulteriori informazioni, consulta Modifica dei gruppi di sicurezza di un'istanza nella Guida per l'utente di Amazon VPC.

Puoi verificare la presenza di un gruppo di sicurezza per il AWS Management Console tuo cluster nella sezione Rete relativa al cluster. In alternativa, puoi farlo usando il seguente AWS CLI comando. Se utilizzi questo comando, sostituisci *my-cluster* con il nome del cluster.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Creazione di un ruolo di esecuzione del Pod Fargate

Quando il cluster si Pods attiva AWS Fargate, i componenti in esecuzione sull'infrastruttura Fargate devono effettuare chiamate alle AWS API per conto dell'utente. Il ruolo di esecuzione del Pod Amazon EKS fornisce le autorizzazioni IAM per eseguire questa operazione. Per creare un ruolo di AWS Fargate Pod esecuzione, consulta. Ruolo IAM per l'esecuzione del Pod Amazon EKS

> **Note**
>
> Se hai creato il cluster `eksctl` utilizzando l'opzione `--fargate`, allora il cluster dispone già di un ruolo di esecuzione del Pod disponibile nella console IAM con il modello `eksctl-`*`my-cluster`*`-FargatePodExecutionRole-`*`ABCDEFGHIJKL`*. Allo stesso modo, se utilizzi `eksctl` per creare i profili Fargate, `eksctl` creerà il ruolo di esecuzione del Pod (se non ne esiste già uno).

Creazione di un profilo Fargate per il cluster

Prima di poter pianificare i Pods in esecuzione su Fargate nel cluster, dovrai definire un profilo Fargate che specifichi i Pods che utilizzano Fargate al momento dell'avvio. Per ulteriori informazioni, consulta AWS Fargate profilo.

> **Note**
>
> Se hai creato il cluster con `eksctl` utilizzando l'opzione `--fargate`, allora un profilo Fargate è già stato creato per il cluster con selettori per tutti i Pods pod negli spazi dei nomi `kube-system` e `default`. Utilizza la procedura seguente per creare profili Fargate per gli altri spazi dei nomi con cui desideri utilizzare Fargate.

È possibile creare un profilo Fargate utilizzando `eksctl` o la AWS Management Console.

eksctl

Questa procedura richiede `eksctl` versione `0.183.0` o successiva. Puoi verificare la versione con il comando seguente:

```
eksctl version
```

Per istruzioni sull'installazione o sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Creazione di un profilo Fargate con **`eksctl`**

Crea il tuo profilo Fargate con il seguente comando `eksctl`, sostituendo ogni *`example value`* con i valori in tuo possesso. È necessario specificare uno spazio dei nomi. Tuttavia, l'opzione `--labels` non è obbligatoria.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

È possibile utilizzare determinati caratteri jolly per *`my-kubernetes-namespace`* e etichette *`key=value`*. Per ulteriori informazioni, consulta Caratteri jolly del profilo Fargate.

AWS Management Console

Per creare un profilo Fargate per un cluster con AWS Management Console

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli il cluster per cui creare un profilo Fargate.
3. Scegli la scheda Calcolo.
4. Nella sezione Profili Fargate, scegli Aggiungi profilo Fargate.
5. Nella pagina Configura il profilo Fargate, procedi come segue:
    a. In Nome, inserisci un nome per il profilo Fargate. Il nome deve essere univoco.
    b. Per Ruolo di esecuzione pod, scegli il ruolo di esecuzione del Pod da utilizzare con il profilo Fargate. Vengono visualizzati solo i ruoli IAM con il principale del servizio

`eks-fargate-pods.amazonaws.com`. Se non è presente alcun ruolo nell'elenco, è necessario crearne uno. Per ulteriori informazioni, consulta Ruolo IAM per l'esecuzione del Pod Amazon EKS.

c. Modifica le sottoreti selezionate in base alle esigenze.

> **Note**
>
> Per i Pods in esecuzione su Fargate sono supportate solo le sottoreti private.

d. In Tag, puoi facoltativamente aggiungere tag al tuo profilo Fargate. Questi tag non si propagano ad altre risorse associate al profilo, ad esempio Pods.

e. Seleziona Successivo.

6. Nella pagina Configura la selezione dei Pod, procedi come segue:

a. In Spazio dei nomi, inserisci uno spazio dei nomi che corrisponda ai Pods.

- È possibile utilizzare spazi dei nomi specifici da abbinare, ad esempio **`kube-system`** o **`default`**.
- È possibile utilizzare determinati caratteri jolly (ad esempio, **`prod-*`**) per abbinare più spazi dei nomi (ad esempio, `prod-deployment` e `prod-test`). Per ulteriori informazioni, consulta Caratteri jolly del profilo Fargate.

b. (Facoltativo) Aggiungi etichette Kubernetes al selettore. In particolare, aggiungile al selettore a cui i Pods nello spazio dei nomi specificato devono corrispondere.

- È possibile aggiungere l'etichetta **`infrastructure: fargate`** al selettore in modo che solo i Pods nello spazio dei nomi specificato che hanno anche l'etichetta `infrastructure: fargate` Kubernetes corrispondano al selettore.
- È possibile utilizzare determinati caratteri jolly (ad esempio, **`key?: value?`**) per abbinare più spazi dei nomi (ad esempio, `keya: valuea` e `keyb: valueb`). Per ulteriori informazioni, consulta la pagina Caratteri jolly del profilo Fargate.

c. Seleziona Next (Successivo).

7. Nella pagina Rivedi e crea, controlla le informazioni relative al profilo Fargate e scegli Crea.

Aggiornamento di CoreDNS

Per impostazione predefinita, CoreDNS è configurato per l'esecuzione sull'infrastruttura Amazon EC2 nei cluster Amazon EKS. Se desideri eseguire i Pods solo su Fargate nel cluster, completa la procedura riportata di seguito.

> **Note**
>
> Se hai creato il cluster `eksctl` utilizzando l'opzione `--fargate`, puoi passare a Passaggi successivi.

1. Crea un profilo Fargate per CoreDNS con il seguente comando. Sostituisci *`my-cluster`* con il nome del cluster, *`111122223333`* con il tuo ID account, *`AmazonEKSFargatePodExecutionRole`* con il nome del ruolo di esecuzione Pod e *`0000000000000001`*, *`0000000000000002`* e *`0000000000000003`* con gli ID delle sottoreti private. Se non disponi di un ruolo di esecuzione del Pod, è necessario prima crearne uno.

   > **Important**
   >
   > L'ARN del ruolo non può includere un percorso diverso da `/`. Ad esempio, se il nome del ruolo è `development/apps/my-role`, è necessario modificarlo in `my-role` quando si specifica l'ARN per tale ruolo. Il formato dell'ARN del ruolo deve essere `arn:aws:iam::111122223333:role/role-name`.

   ```
   aws eks create-fargate-profile \
       --fargate-profile-name coredns \
       --cluster-name my-cluster \
       --pod-execution-role-arn
    arn:aws:iam::111122223333:role/AmazonEKSFargatePodExecutionRole \
       --selectors namespace=kube-system,labels={k8s-app=kube-dns} \
       --subnets subnet-0000000000000001 subnet-0000000000000002
    subnet-0000000000000003
   ```

2. Utilizza il comando seguente per rimuovere l'annotazione `eks.amazonaws.com/compute-type : ec2` dai CoreDNS Pods.

   ```
   kubectl patch deployment coredns \
   ```

```
    -n kube-system \
    --type json \
    -p='[{"op": "remove", "path": "/spec/template/metadata/annotations/
eks.amazonaws.com~1compute-type"}]'
```

Passaggi successivi

- Puoi avviare la migrazione delle applicazioni esistenti per l'esecuzione su Fargate con il seguente flusso di lavoro.
  1. Crea un profilo Fargate che corrisponda allo spazio dei nomi Kubernetes e alle etichette Kubernetes dell'applicazione.
  2. Elimina e crea nuovamente tutti i Pods esistenti in modo che siano programmati su Fargate. Ad esempio, il comando seguente attiva un rollout dell'implementazione `coredns`. Puoi modificare lo spazio dei nomi e il tipo di implementazione per aggiornare Pods specifici.

     ```
     kubectl rollout restart -n kube-system deployment coredns
     ```
- Implementa Bilanciamento del carico di applicazione su Amazon EKS per permettere l'esecuzione di oggetti Ingress per i Pods in esecuzione su Fargate.
- Puoi utilizzare Vertical Pod Autoscaler per impostare la dimensione corretta iniziale della CPU e la memoria per i Pods Fargate, quindi utilizzare il comando Horizontal Pod Autoscaler per dimensionare tali Pods. Se desideri che il Vertical Pod Autoscaler implementi di nuovo automaticamente i Pods su Fargate con combinazioni di CPU e memoria più grandi, impostare la modalità Autoscaler del Pod Vertical su `Auto` o `Recreate`. Ciò garantisce una corretta funzionalità. Per ulteriori informazioni, consulta la documentazione di Vertical Pod Autoscaler su GitHub.
- Ora puoi impostare il collector AWS Distro per OpenTelemetry (ADOT) per il monitoraggio delle applicazioni seguendo queste istruzioni.

AWS Fargate profilo

> **Important**
>
> AWS Fargate con Amazon EKS non è disponibile negli AWS GovCloud Stati Uniti orientali e AWS GovCloud negli Stati Uniti occidentali.

Prima di programmare i Pods su Fargate nel cluster, è necessario definire almeno un profilo Fargate che specifichi i Pods utilizzati da Fargate al momento dell'avvio.

Come amministratore, puoi utilizzare il profilo Fargate per dichiarare quali profili Pods vengono eseguiti su Fargate. Puoi effettuare questa operazione tramite i selettori del profilo. Puoi aggiungere fino a cinque selettori a ogni profilo. Ogni selettore deve contenere uno spazio dei nomi. Il selettore può includere anche etichette. Il campo etichetta è costituito da più coppie chiave-valore facoltative. I pod che corrispondono ai selettori sono programmati su Fargate. I pod vengono abbinati utilizzando uno spazio dei nomi e le etichette specificate nel selettore. Se un selettore dello spazio dei nomi è definito senza etichette, Amazon EKS tenterà di pianificare tutti i Pods che vengono eseguiti in tale spazio dei nomi su Fargate utilizzando il profilo. Se a to-be-scheduled Pod corrisponde a uno qualsiasi dei selettori nel profilo Fargate, allora è programmato su Pod Fargate.

Se Pod corrisponde a più profili Fargate, puoi specificare quale profilo a Pod utilizza aggiungendo la seguente etichetta Kubernetes alla Pod specifica: `eks.amazonaws.com/fargate-profile: my-fargate-profile`. Il Pod dovrà corrispondere a un selettore in quel profilo per essere pianificato su Fargate. Le regole di affinità/anti-affinità Kubernetes non vengono prese in considerazione e non sono necessarie con i Pods di Amazon EKS Fargate.

Quando crei un profilo Fargate, devi specificare un ruolo di esecuzione Pod. Questo ruolo di esecuzione è per i componenti Amazon EKS che vengono eseguiti su infrastruttura Fargate utilizzando il profilo. Viene aggiunto al controllo degli accessi basato sul ruolo (RBAC) di Kubernetes del cluster per l'autorizzazione. Ciò consente a `kubelet` in esecuzione sull'infrastruttura Fargate di registrarsi con il cluster Amazon EKS in modo che possa essere visualizzato nel cluster come nodo. Il ruolo di esecuzione del Pod fornisce anche le autorizzazioni IAM per l'infrastruttura Fargate per consentire l'accesso in lettura ai repository di immagini di Amazon ECR. Per ulteriori informazioni, consulta Ruolo IAM per l'esecuzione del Pod Amazon EKS.

I profili Fargate non possono essere modificati. Tuttavia, è possibile creare un nuovo profilo aggiornato per sostituire un profilo esistente e quindi eliminare l'originale.

> **Note**
>
> Qualsiasi Pods in esecuzione utilizzando un profilo Fargate vengono arrestati e messi in sospeso quando il profilo viene eliminato.

Se tutti i profili in un cluster sono nello stato `DELETING`, è necessario attendere che tale profilo Fargate finisca l'eliminazione prima di poter creare altri profili in tale cluster.

Amazon EKS e Fargate cercano di implementare i Pods su ciascuna delle sottoreti definite nel profilo Fargate. Tuttavia, potresti ritrovarti con una diffusione irregolare. Se devi avere una diffusione uniforme, usa due profili Fargate. Anche la diffusione è importante in scenari in cui si desidera implementare due repliche e non si vogliono tempi di inattività. È consigliabile che ogni profilo abbia una sola sottorete.

Componenti del profilo Fargate

Un profilo Fargate contiene i componenti elencati di seguito.

Ruolo di esecuzione del pod

Quando il cluster si Pods attiva AWS Fargate, il `kubelet` cluster in esecuzione sull'infrastruttura Fargate deve effettuare chiamate alle AWS API per conto dell'utente. Ad esempio, deve effettuare chiamate per estrarre le immagini del container da Amazon ECR. Il ruolo di esecuzione del Pod Amazon EKS fornisce le autorizzazioni IAM per eseguire questa operazione.

Quando crei un profilo Fargate, devi specificare un ruolo di esecuzione del Pod da utilizzare con i tuoi Pods. Questo ruolo viene aggiunto al controllo degli accessi basato sul ruolo (RBAC) di Kubernetes del cluster per l'autorizzazione. Ciò consente a `kubelet` in esecuzione sull'infrastruttura Fargate di registrarsi con il cluster Amazon EKS in modo che possa essere visualizzato nel cluster come nodo. Per ulteriori informazioni, consulta Ruolo IAM per l'esecuzione del Pod Amazon EKS.

Sottoreti

Gli ID delle sottoreti per avviare i Pods che utilizzano questo profilo. In questo momento, ai Pods in esecuzione su Fargate non vengono assegnati indirizzi IP pubblici. Di conseguenza, in virtù di questo parametro, sono accettate solo le sottoreti private senza routing diretto a un gateway Internet.

Selettori

I selettori da abbinare ai Pods per utilizzare questo profilo Fargate. È possibile specificare fino a cinque selettori in un profilo Fargate. I selettori hanno le seguenti componenti:

- Spazio dei nomi: specifica uno spazio dei nomi per un selettore. Il selettore corrisponde solo a Pods che vengono creati in questo spazio dei nomi. Tuttavia, è possibile creare più selettori per rivolgersi a più spazi dei nomi.
- Etichette: è possibile specificare facoltativamente le etichette Kubernetes che corrispondono al selettore. Il selettore corrisponde solo ai Pods che hanno tutte le etichette specificate nel selettore.

Caratteri jolly del profilo Fargate

Oltre ai caratteri consentiti da Kubernetes, puoi utilizzare **`*`** e **`?`** nei criteri di selezione per gli spazi dei nomi, chiavi di etichette e valori di etichette:

- **`*`** rappresenta nessuno, uno o più caratteri. Ad esempio, **`prod*`** può rappresentare `prod` e `prod-metrics`.
- **`?`** rappresenta un singolo carattere (ad esempio, **`value?`** può rappresentare `valuea`). Tuttavia, non può rappresentare `value` e `value-a`, perché `?` può rappresentare esattamente un solo carattere.

Questi caratteri jolly possono essere utilizzati in qualsiasi posizione e in combinazione (ad esempio **`prod*`**, **`*dev`**, e **`frontend*?`**). Altri caratteri jolly e forme di corrispondenza del modello, ad esempio le espressioni regolari, non sono supportati.

Se sono presenti più profili corrispondenti per lo spazio dei nomi e le etichette nelle specifiche del Pod, Fargate seleziona il profilo in base all'ordinamento alfanumerico e al nome del profilo. Ad esempio, se entrambi i profili A (con il nome `beta-workload`) e il profilo B (con il nome `prod-workload`) hanno dei selettori corrispondenti per i Pods da avviare, Fargate sceglie il profilo A (`beta-workload`) per i Pods. I Pods hanno etichette con il profilo A sui Pods (ad esempio, `eks.amazonaws.com/fargate-profile=beta-workload`).

Se desideri migrare i Pods Fargate esistenti su nuovi profili che utilizzano caratteri jolly, puoi farlo in due modi:

- Crea un nuovo profilo con i selettori corrispondenti e poi elimina i vecchi profili. I pod etichettati con vecchi profili vengono riprogrammati con nuovi profili corrispondenti.
- Se desideri migrare i carichi di lavoro ma non hai la certezza di quali siano le etichette Fargate su ogni Pod Fargate, puoi utilizzare il seguente metodo. Crea un nuovo profilo con un nome che risulti primo in ordine alfanumerico tra i profili dello stesso cluster. Quindi, ricicla i Pods Fargate che devono essere migrati a nuovi profili.

Creazione di un profilo Fargate

Questo argomento descrive come creare un profilo Fargate. È inoltre necessario aver creato un ruolo di esecuzione del Pod da utilizzare per il profilo Fargate. Per ulteriori informazioni, vedere. Ruolo IAM per l'esecuzione del Pod Amazon EKS Podsche sono in esecuzione su Fargate sono supportati solo

su sottoreti private con accesso gateway NAT Servizi AWS, ma non tramite un percorso diretto verso un Internet Gateway. Ciò implica che il VPC del cluster debba avere a disposizione sottoreti private. È possibile creare un profilo con `eksctl` o la AWS Management Console.

Questa procedura richiede `eksctl` versione `0.183.0` o successiva. Puoi verificare la versione con il comando seguente:

```
eksctl version
```

Per istruzioni sull'installazione o sull'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

eksctl

Creazione di un profilo Fargate con **`eksctl`**

Crea il tuo profilo Fargate con il seguente comando `eksctl`, sostituendo ogni *`example value`* con i valori in tuo possesso. È necessario specificare uno spazio dei nomi. Tuttavia, l'opzione `--labels` non è obbligatoria.

```
eksctl create fargateprofile \
    --cluster my-cluster \
    --name my-fargate-profile \
    --namespace my-kubernetes-namespace \
    --labels key=value
```

È possibile utilizzare determinati caratteri jolly per *`my-kubernetes-namespace`* e etichette *`key=value`*. Per ulteriori informazioni, consulta Caratteri jolly del profilo Fargate.

AWS Management Console

Per creare un profilo Fargate per un cluster con AWS Management Console

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli il cluster per cui creare un profilo Fargate.
3. Scegli la scheda Calcolo.
4. Nella sezione Profili Fargate, scegli Aggiungi profilo Fargate.
5. Nella pagina Configure Fargate profile (Configura profilo Fargate), procedere come segue:

a. In Nome, inserisci un nome univoco per il profilo Fargate, ad esempio ***my-profile***.

b. Per il ruolo di esecuzione del pod, scegli il ruolo di esecuzione del Pod da utilizzare con il profilo Fargate. Vengono visualizzati solo i ruoli IAM con il principale del servizio `eks-fargate-pods.amazonaws.com`. Se non è presente alcun ruolo nell'elenco, è necessario crearne uno. Per ulteriori informazioni, consulta Ruolo IAM per l'esecuzione del Pod Amazon EKS.

c. Modifica le sottoreti selezionate in base alle esigenze.

> **Note**
>
> Per i Pods in esecuzione su Fargate sono supportate solo le sottoreti private.

d. In Tag, puoi facoltativamente aggiungere tag al tuo profilo Fargate. Questi tag non si propagano ad altre risorse associate al profilo, ad esempio i suoi Pods.

e. Seleziona Successivo.

6. Nella pagina Configura la selezione dei Pod, procedi come segue:

a. In Spazio dei nomi, inserisci uno spazio dei nomi che corrisponda ai Pods.

- È possibile utilizzare spazi dei nomi specifici da abbinare, ad esempio **`kube-system`** o **`default`**.
- È possibile utilizzare determinati caratteri jolly (ad esempio, **`prod-*`**) per abbinare più spazi dei nomi (ad esempio, `prod-deployment` e `prod-test`). Per ulteriori informazioni, consulta Caratteri jolly del profilo Fargate.

b. (Facoltativo) Aggiungi etichette Kubernetes al selettore. In particolare, aggiungerle al selettore con cui i Pods nello spazio dei nomi specificato devono corrispondere.

- È possibile aggiungere l'etichetta **`infrastructure: fargate`** al selettore in modo che solo i Pods nello spazio dei nomi specificato che hanno anche l'etichetta `infrastructure: fargate` Kubernetes corrispondano al selettore.
- È possibile utilizzare determinati caratteri jolly (ad esempio, **`key?: value?`**) per abbinare più spazi dei nomi (ad esempio, `keya: valuea` e `keyb: valueb`). Per ulteriori informazioni, consulta la pagina Caratteri jolly del profilo Fargate.

c. Seleziona Next (Successivo).

7. Nella pagina Rivedi e crea, controlla le informazioni relative al profilo Fargate e scegli Crea.

Eliminazione di un profilo Fargate

Questo argomento descrive come eliminare un profilo Fargate.

Quando elimini un profilo Fargate, tutti i Pods che sono stati programmati su Fargate con quel profilo vengono eliminati. Se questi Pods corrispondono a un altro profilo Fargate, vengono allora programmati su Fargate con quel profilo. Se non corrispondono più a nessun profilo Fargate, non saranno programmati su Fargate e potrebbero rimanere in sospeso.

Solo un profilo Fargate in un cluster alla volta può essere nello stato `DELETING`. Devi attendere il completamento dell'eliminazione di un profilo Fargate prima di poter eliminare altri profili nel cluster.

È possibile eliminare un profilo con`eksctl`, il AWS Management Console, o il AWS CLI. Seleziona la scheda con il nome dello strumento che desideri utilizzare per eliminare il profilo.

eksctl

Eliminazione di un profilo Fargate con **`eksctl`**

Utilizza il comando seguente per eliminare un profilo da un cluster. Sostituisci ogni *`example value`* con i valori in tuo possesso.

```
eksctl delete fargateprofile  --name my-profile --cluster my-cluster
```

AWS Management Console

Per eliminare un profilo Fargate da un cluster con AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, seleziona Cluster. Nell'elenco dei cluster, scegli il cluster da cui desideri eliminare il profilo Fargate.
3. Scegli la scheda Calcolo.
4. Scegli il profilo Fargate da eliminare e quindi Elimina.
5. Nella pagina Elimina profilo Fargate, digita il nome del profilo e scegli Elimina.

AWS CLI

Eliminazione di un profilo Fargate con AWS CLI

Utilizza il comando seguente per eliminare un profilo da un cluster. Sostituisci ogni *example value* con i valori in tuo possesso.

```
aws eks delete-fargate-profile --fargate-profile-name my-profile --cluster-name my-cluster
```

Configurazione dei Pod Fargate

> **Important**
>
> AWS Fargatecon Amazon EKS non è disponibile negli AWS GovCloud Stati Uniti orientali e AWS GovCloud negli Stati Uniti occidentali.

In questa sezione sono descritti alcuni dettagli di configurazione dei Pod univoci per l'esecuzione dei pod Kubernetes Pods su AWS Fargate.

Pod CPU e memoria

Con Kubernetes, è possibile definire le richieste, una quantità minima di vCPU e le risorse di memoria allocate a ciascun container in unPod.Podssono programmati da Kubernetes per garantire che almeno le risorse richieste per ciascun Pod siano disponibili nella risorsa di calcolo. Per ulteriori informazioni, consulta Gestione delle risorse di calcolo per i container nella documentazione di Kubernetes.

> **Note**
>
> Poiché Amazon EKS Fargate esegue solo un Pod per nodo, lo scenario di espulsione dei Pods in caso di meno risorse non si verifica. Tutti i Pods Fargate di Amazon EKS funzionano con priorità garantita, quindi la CPU e la memoria richieste devono essere uguali al limite per tutti i container. Per ulteriori informazioni, consulta Configure Quality of Service for PodsPods (Configurare la qualità di servizio per i pod) nella documentazione di Kubernetes.

Quando i Pods sono programmati su Fargate, le prenotazioni di vCPU e memoria all'interno della specifica del Pod determinano la quantità di CPU e memoria per effettuare il provisioning per il Pod.

- La richiesta massima di qualsiasi container Init viene utilizzata per determinare i requisiti di memoria e vCPU della richiesta Init.
- Le richieste per tutti i container con esecuzione prolungata vengono aggiunte per determinare i requisiti di memoria e vCPU della richiesta con esecuzione prolungata.
- Il valore maggiore tra i due valori sopra indicati viene scelto per la richiesta di vCPU e memoria da utilizzare per il Pod.
- Fargate aggiunge 256 MB alla prenotazione di memoria di ogni Pod per i componenti Kubernetes richiesti (`kubelet`, `kube-proxy` e `containerd`).

Fargate arrotonda alla configurazione di calcolo mostrata di seguito che corrisponde più strettamente alla somma delle richieste di vCPU e di memoria, al fine di garantire ai Pods sempre le risorse necessarie per la loro esecuzione.

Se non si specifica una combinazione di vCPU e memoria, viene utilizzata la combinazione più piccola disponibile (.25 vCPU e 0,5 GB di memoria).

La tabella seguente mostra le combinazioni di vCPU e memoria disponibili per i Pods in esecuzione su Fargate.

Valore vCPU	Valore memoria
.25 vCPU	0,5 GB, 1 GB, 2 GB
.5 vCPU	1 GB, 2 GB, 3 GB, 4 GB
1 vCPU	2 GB, 3 GB, 4 GB, 5 GB, 6 GB, 7 GB, 8 GB
2 vCPU	Tra 4 GB e 16 GB in incrementi di 1 GB
4 vCPU	Tra 8 GB e 30 GB in incrementi di 1 GB
8 vCPU	Tra 16 GB e 60 GB in incrementi di 4 GB
16 vCPU	Tra 32 GB e 120 GB in incrementi di 8 GB

La memoria aggiuntiva riservata ai componenti Kubernetes può causare un'attività Fargate con più vCPUs di quelle richieste per il provisioning. Ad esempio, una richiesta per 1 vCPU e 8 GB di

memoria avrà 256 MB aggiunti alla richiesta di memoria e fornirà un'attività Fargate di 2 vCPU e 9 GB di memoria, in quanto non è disponibile alcuna attività con 1 vCPU e 9 GB di memoria.

Non vi è alcuna correlazione tra la dimensione del Pod in esecuzione su Fargate e la dimensione del nodo riportata da Kubernetes con `kubectl get nodes`. La dimensione del nodo riportata è spesso maggiore della capacità del Pod. È possibile verificare la capacità del Pod utilizzando il comando seguente. Sostituisci *`default`* con il nome del Pod e *`pod-name`* con il nome dello spazio dei nomi di Pod.

```
kubectl describe pod --namespace default pod-name
```

Di seguito viene riportato un output di esempio:

```
[...]
annotations:
    CapacityProvisioned: 0.25vCPU 0.5GB
[...]
```

L'annotazione `CapacityProvisioned` rappresenta la capacità del Pod applicata e determina il costo del Pod in esecuzione su Fargate. Per informazioni sui prezzi di queste configurazioni di calcolo, consulta Prezzi di AWS Fargate.

Archiviazione in Fargate

Un Pod in esecuzione su Fargate monta automaticamente un file system Amazon EFS. Non è possibile utilizzare il provisioning dinamico dei volumi persistenti con nodi Fargate, ma è possibile utilizzare il provisioning statico. Per ulteriori informazioni, consulta Amazon EFS CSI Driver on GitHub.

Una volta fornito, ciascun Pod in esecuzione su Fargate riceve uno storage temporaneo predefinito di 20 GiB. Questo tipo di archiviazione viene eliminato dopo che un Pod si ferma. Per i nuovi Pods avviati su Fargate, la crittografia del volume di archiviazione temporanea è abilitata per impostazione predefinita. L'archiviazione temporanea dei Pod viene crittografata con un algoritmo di crittografia AES-256 utilizzando chiavi gestite da AWS Fargate.

> **Note**
>
> Lo storage utilizzabile predefinito per Amazon EKSPods che funzionano su Fargate sono inferiori a 20 GiB. Questo perché parte dello spazio viene utilizzato da`kubelet` e altri moduli Kubernetes che vengono caricati all'interno di Pod.

Puoi aumentare la quantità totale di storage temporaneo fino a un massimo di 175 GiB. Per configurare la dimensione con Kubernetes, specificare le richieste della risorsa `ephemeral-storage` per ogni container in un Pod. Quando Kubernetes pianifica Pods, garantisce che la somma delle richieste di risorse per ciascun Pod sia inferiore alla capacità del task Fargate. Per ulteriori informazioni, consulta Gestione delle risorse per pod e containerPods nella documentazione di Kubernetes.

Amazon EKS Fargate fornisce un maggiore spazio di archiviazione temporaneo di quello richiesto ai fini dell'utilizzo del sistema. Ad esempio, una richiesta di 100 GiB fornirà un'attività Fargate con 115 GiB di storage temporaneo.

Applicazione di patch del sistema operativo Fargate

> **Important**
>
> AWS Fargatecon Amazon EKS non è disponibile negli AWS GovCloud Stati Uniti orientali e AWS GovCloud negli Stati Uniti occidentali.

Amazon EKS applica periodicamente delle patch al sistema operativo per i nodi AWS Fargate al fine di garantirne la sicurezza. Come parte del processo di applicazione delle patch, ricicliamo i nodi per installare le patch del sistema operativo. Gli aggiornamenti vengono effettuati in modo da avere il minore impatto possibile sui servizi. Tuttavia, se i Pods non vengono espulsi correttamente, a volte è necessario eliminarli. Di seguito sono riportate le azioni che è possibile intraprendere per ridurre al minimo le potenziali interruzioni:

- Imposta i budget di interruzione dei Pod (PDB) appropriati per limitare il numero di Pods inattivi simultaneamente.
- Crea EventBridge regole Amazon per gestire gli sfratti falliti prima che Pods vengano eliminati.
- Crea una configurazione di notifica in Notifiche utente di AWS.

Lavora a stretto contatto con la community Kubernetes per rendere disponibili il più rapidamente possibile le correzioni dei bug e le patch di sicurezza. L'avvio di tutti i Pods Fargate avviene con la versione più recente della patch Kubernetes, disponibile da Amazon EKS per la versione Kubernetes del cluster. Se disponi di un Pod con una patch di una versione precedente, Amazon EKS potrebbe riciclarlo per aggiornarlo alla versione più recente. Ciò consente di applicare ai Pods gli aggiornamenti per la sicurezza più recenti, In tal modo, se si verifica una problematica Common Vulnerabilities and Exposures (CVE), riceverai aggiornamenti costanti per ridurre i rischi per la sicurezza.

Per limitare il numero di Pods inattivi durante il riciclo dei Pods, puoi impostare i budget di interruzione dei Pod (PDB). Con questa opzione è possibile specificare la disponibilità minima in base ai requisiti di ciascuna applicazione e consentire, allo stesso tempo, l'esecuzione di aggiornamenti. Per ulteriori informazioni, consulta Specifica di un budget di interruzione per l'applicazione nella Documentazione di Kubernetes.

Amazon EKS utilizza l'API di espulsione per interrompere in modo sicuro il Pod rispettando i PDB impostati per l'applicazione. I pod vengono espulsi dalla zona di disponibilità per ridurre al minimo l'impatto. Se l'espulsione ha esito positivo, il nuovo Pod riceve la patch più recente e non sono richieste ulteriori operazioni.

Se l'espulsione di un Pod ha esito negativo, Amazon EKS invia al tuo account un evento con dettagli relativi al Pods in cui si è verificato l'errore. È possibile intraprendere alcune azioni in merito al messaggio prima del periodo di interruzione programmato, che varia in base all'urgenza della patch. Quando è il momento, Amazon EKS proverà a espellere nuovamente i Pods. Questa volta, tuttavia, se l'espulsione non riesce, non viene inviato un nuovo evento. Se l'espulsione continua a non riuscire, i Pods esistenti verranno eliminati periodicamente in modo che i nuovi Pods possano disporre della patch più recente.

L'esempio seguente mostra un evento ricevuto in caso di espulsione del Pod che non riesce. Il messaggio contiene dettagli sul cluster, il nome del Pod, lo spazio dei nomi del Pod, il profilo Fargate e il periodo di interruzione pianificato.

```
{
    "version": "0",
    "id": "12345678-90ab-cdef-0123-4567890abcde",
    "detail-type": "EKS Fargate Pod Scheduled Termination",
    "source": "aws.eks",
    "account": "111122223333",
    "time": "2021-06-27T12:52:44Z",
```

```
    "region": "region-code",
    "resources": [
        "default/my-database-deployment"
    ],
    "detail": {
        "clusterName": "my-cluster",
        "fargateProfileName": "my-fargate-profile",
        "podName": "my-pod-name",
        "podNamespace": "default",
        "evictErrorMessage": "Cannot evict pod as it would violate the pod's disruption
 budget",
        "scheduledTerminationTime": "2021-06-30T12:52:44.832Z[UTC]"
    }
}
```

L'associazione di molteplici budget di interruzione dei pod (PDB) a un singolo Pod può causare un evento di errore di espulsione. Questo evento restituisce il messaggio di errore seguente.

```
"evictErrorMessage": "This pod has multiple PodDisruptionBudget, which the eviction
 subresource does not support",
```

Puoi creare un'azione desiderata in base a questo evento. Ad esempio, puoi regolare il budget di interruzione dei Pod (PDB) per controllare il modo in cui i Pods vengono espulsi. Più specificamente, si supponga di iniziare con un PDB che specifica la percentuale di destinazione di Pods disponibili. Prima che i Pods vengano terminati in modo forzato durante un aggiornamento, puoi regolare il PDB su una percentuale di Pods differente. Per ricevere questo evento, devi creare una EventBridge regola Amazon nel Account AWS e a Regione AWS cui appartiene il cluster. La regola deve utilizzare il modello personalizzato seguente. Per ulteriori informazioni, consulta la sezione Creazione di EventBridge regole Amazon che reagiscono agli eventi nella Amazon EventBridge User Guide.

```
{
  "source": ["aws.eks"],
  "detail-type": ["EKS Fargate Pod Scheduled Termination"]
}
```

L'evento può essere configurato per acquisire un obiettivo adeguato impostato dall'utente. Per un elenco completo degli obiettivi disponibili, consulta Amazon EventBridge targets nella Amazon EventBridge User Guide. Inoltre puoi creare una configurazione di notifica in Notifiche utente AWS. Quando usi la AWS Management Console per creare la notifica, sotto la voce Regole degli eventi scegli Elastic Kubernetes Service (EKS) per il Nome di Servizio AWS e Arresto programmato di EKS

Fargate Pod per il Tipo di evento. Per ulteriori informazioni, consulta Guida introduttiva alle Notifiche utente AWS nella Guida per l'utente delle Notifiche utente AWS.

Parametri Fargate

> **Important**
>
> AWS Fargate con Amazon EKS non è disponibile nelle regioni AWS GovCloud (Stati Uniti-Est) e AWS GovCloud (Stati Uniti-Ovest).

Puoi raccogliere parametri di sistema e di utilizzo di CloudWatch per AWS Fargate.

Parametri di applicazione

Per le applicazioni in esecuzione su Amazon EKS e AWS Fargate puoi utilizzareAWS Distro for OpenTelemetry (ADOT). ADOT consente di raccogliere i parametri di sistema e di inviarli ai pannelli di controllo di Approfondimenti sui container CloudWatch. Per iniziare a utilizzare ADOT per le applicazioni in esecuzione su Fargate, consulta Utilizzo di CloudWatch Container Insights con AWS Distro for OpenTelemetry nella documentazione di ADOT.

Parametri di utilizzo

Puoi utilizzare i parametri di utilizzo di CloudWatch per fornire visibilità sull'utilizzo delle risorse del tuo account. Utilizza questi parametri per visualizzare l'uso del servizio corrente su grafici e pannelli di controllo di CloudWatch.

I parametri di utilizzo AWS Fargate corrispondono alle quote di servizio AWS. È possibile configurare gli allarmi che avvisano quando l'uso si avvicina a una quota di servizio. Per ulteriori informazioni sulle quote di servizio per Fargate, consulta Service Quotas di Amazon EKS.

AWS Fargate pubblica i seguenti parametri nello spazio dei nomi `AWS/Usage`.

Parametro	Descrizione
`ResourceCount`	Il numero totale delle risorse specificate in esecuzione nell'account. La risorsa è definita dalle dimensioni associate attraverso il parametro.

Le seguenti dimensioni vengono utilizzate per perfezionare i parametri di utilizzo pubblicati da AWS Fargate.

Dimensione	Descrizione
`Service`	Il nome del servizio AWS contenente la risorsa. Per i parametri di utilizzo di AWS Fargate, il valore di questa dimensione è `Fargate`.
`Type`	Il tipo di entità che viene segnalato. Attualmente, l'unico valore valido per i parametri di utilizzo AWS Fargate è `Resource`.
`Resource`	Il tipo di risorsa in esecuzione. Attualmente, AWS Fargate restituisce informazioni sull'utilizzo on demand di Fargate. Il valore della risorsa per l'utilizzo on demand di Fargate è `OnDemand`. **Note** L'utilizzo on demand di Fargate combina i Pods Amazon EKS utilizzando Fargate, le attività Amazon ECS che utilizzano il tipo di avvio di Fargate e le attività Amazon ECS che usano il provider di capacità `FARGATE`.
`Class`	La classe della risorsa monitorata. Attualmente, AWS Fargate non utilizza la dimensione della classe.

Creazione di un allarme CloudWatch per monitorare i parametri di utilizzo delle risorse di Fargate

AWS Fargate fornisce i parametri di utilizzo di CloudWatch che corrispondono alle quote di servizio AWS per l'utilizzo di risorse on demand di Fargate. Nella console Service Quotas (Quote di Servizio) è possibile visualizzare l'utilizzo in un grafico. È inoltre possibile configurare gli allarmi che avvisano quando l'uso si avvicina a una quota di servizio. Per ulteriori informazioni, consulta Parametri Fargate.

Attieniti alla seguente procedura per creare un allarme CloudWatch basato sui parametri di utilizzo delle risorse Fargate.

Per creare un allarme basato sulle quote di utilizzo Fargate (AWS Management Console)

1. Apri la console Service Quotas all'indirizzo https://console.aws.amazon.com/servicequotas/
2. Nel pannello di navigazione a sinistra, scegli Servizi AWS.
3. Nell'elenco Servizi AWS, cerca e seleziona AWS Fargate.
4. Nell'elenco Service Quotas, seleziona la quota di utilizzo Fargate per cui desideri creare un allarme.
5. Nella sezione Amazon CloudWatch alarms (Allarmi Amazon CloudWatch), seleziona Create (Crea).
6. Per Soglia di allarme, scegli la percentuale del valore della quota applicata che desideri impostare come valore per l'allarme.
7. Per Nome allarme, immetti un nome per l'allarme e quindi scegli Crea.

Logging di Fargate

> **Important**
>
> AWS Fargate con Amazon EKS non è disponibile negli AWS GovCloud Stati Uniti orientali e AWS GovCloud negli Stati Uniti occidentali.

Amazon EKS su Fargate offre un router di log integrato basato su Fluent Bit. Ciò significa che non esegui esplicitamente un container Fluent Bit come sidecar, ma Amazon lo gestisce per te. Tutto quello che devi fare è configurare il router di log. La configurazione avviene attraverso un `ConfigMap` dedicato che deve soddisfare i seguenti criteri:

- Deve essere denominato `aws-logging`
- Creato in uno spazio dei nomi dedicato denominato `aws-observability`
- Non può contenere più di 5.300 caratteri.

Una volta creato il `ConfigMap`, Amazon EKS su Fargate lo rileva in automatico e con esso configura il router di log. Fargate utilizza una versione di AWS forFluent Bit, una distribuzione conforme a upstream di managed by. Fluent Bit AWS Per ulteriori informazioni, vedere for on. AWSFluent Bit GitHub

Il log router consente di utilizzare l'ampia gamma di servizi AWS per l'analisi e l'archiviazione dei log. Puoi trasmettere i log da Fargate direttamente ad Amazon CloudWatch, Amazon OpenSearch Service. Puoi anche trasmettere i log verso destinazioni come Amazon S3, AmazonKinesis Data Streams e strumenti partner tramite Amazon Data Firehose.

Prerequisiti

- Un profilo Fargate esistente che specifica uno spazio dei nomi Kubernetes esistente in cui si implementano i Pods Fargate. Per ulteriori informazioni, consulta Creazione di un profilo Fargate per il cluster.
- Il ruolo di esecuzione del Pod Fargate esistente. Per ulteriori informazioni, consulta Creazione di un ruolo di esecuzione del Pod Fargate.

Configurazione del router di log

Configurazione del router di log

Nelle fasi seguenti, sostituisci *example value* con i valori in tuo possesso.

1. Crea uno spazio dei nomi Kubernetes dedicato denominato `aws-observability`.

   a. Salva nel tuo computer i seguenti contenuti in un file denominato *aws-observability-namespace*`.yaml`. Il valore per `name` deve essere `aws-observability` e l'etichetta `aws-observability: enabled` è obbligatoria.

   ```
   kind: Namespace
   apiVersion: v1
   metadata:
     name: aws-observability
     labels:
       aws-observability: enabled
   ```

   b. Crea lo spazio dei nomi.

   ```
   kubectl apply -f aws-observability-namespace.yaml
   ```

2. Crea una `ConfigMap` con valore dei dati `Fluent Conf` per spedire i log del container verso una destinazione. Fluent Conf è Fluent Bit, un linguaggio di configurazione del processore di log veloce e leggero utilizzato per instradare i registri del container verso una destinazione di registro

selezionata. Per ulteriori informazioni, consulta File di configurazione nella documentazione di Fluent Bit.

> **Important**
>
> In una `Fluent Conf` tipica, le sezioni principali incluse sono `Service`, `Input`, `Filter` e `Output`. Tuttavia, il router di log di Fargate accetta solo:
>
> - Le sezioni `Filter` e `Output`.
> - Una sezione `Parser`.
>
> Se fornisci altre sezioni, queste verranno rifiutate.

Il router di log Fargate gestisce le sezioni `Service` e `Input`. Ha la sezione `Input` seguente, che non può essere modificata e non è necessaria nel tuo `ConfigMap`. Tuttavia, è possibile ricavarne informazioni, ad esempio il limite del buffer di memoria e il tag applicato per i log.

```
[INPUT]
    Name tail
    Buffer_Max_Size 66KB
    DB /var/log/flb_kube.db
    Mem_Buf_Limit 45MB
    Path /var/log/containers/*.log
    Read_From_Head On
    Refresh_Interval 10
    Rotate_Wait 30
    Skip_Long_Lines On
    Tag kube.*
```

Quando crei la `ConfigMap`, ricorda le seguenti regole utilizzate da Fargate per convalidare i campi:

- `[FILTER]`, `[OUTPUT]`, e `[PARSER]` dovrebbero essere specificati sotto ogni chiave corrispondente. Ad esempio: `[FILTER]` deve essere in `filters.conf`. È possibile avere uno o più `[FILTER]` in `filters.conf`. Anche le sezioni `[OUTPUT]` e `[PARSER]` dovrebbero essere specificate sotto le chiavi corrispondenti. Specificando più sezioni `[OUTPUT]`, è possibile instradare i registritha a destinazioni diverse contemporaneamente.

- Fargate convalida le chiavi richieste per ogni sezione. `Name` e `match` sono necessari per ogni `[FILTER]` e `[OUTPUT]`. `Name` e `format` sono necessari per ogni `[PARSER]`. Le chiavi non fanno distinzione tra maiuscole e minuscole.
- Variabili di ambiente come `${ENV_VAR}` non sono ammesse nella `ConfigMap`.
- La rientranza deve essere la stessa sia per la direttiva che per la coppia chiave-valore all'interno di ogni `filters.conf`, `output.conf` e `parsers.conf`. Le coppie chiave-valore devono essere rientranti piuttosto che direttive.
- Fargate effettua la convalida in base ai seguenti filtri supportati: `grep`, `parser`, `record_modifier`, `rewrite_tag`, `throttle`, `nest`, `modify` e `kubernetes`.
- Fargate effettua la convalida in base al seguente output supportato: `es`, `firehose`, `kinesis_firehose`, `cloudwatch`, `cloudwatch_logs`, e `kinesis`.
- Nella `ConfigMap` deve essere fornito almeno un plug-in `Output` supportato per abilitare il logging. `Filter` e `Parser` non sono necessari per abilitare il logging.

È possibile anche eseguire Fluent Bit su Amazon EC2 utilizzando la configurazione desiderata per risolvere eventuali problemi derivanti dalla convalida. Crea il `ConfigMap` utilizzando uno degli esempi seguenti.

> **Important**
>
> Il logging Fargate di Amazon EKS non supporta la configurazione dinamica di `ConfigMaps`. Eventuali modifiche a `ConfigMaps` vengono applicate solo ai nuovi Pods. Le modifiche non vengono applicate ai Pods esistenti.

Crea un `ConfigMap` utilizzando l'esempio per la destinazione di log desiderata.

> **Note**
>
> Puoi anche utilizzare Flusso di dati Amazon Kinesis come destinazione dei log. Se utilizzi Flusso di dati Kinesis, assicurati che al ruolo di esecuzione del pod sia stata concessa l'autorizzazione `kinesis:PutRecords`. Per ulteriori informazioni, consulta la sezione Autorizzazioni di Flusso di dati Amazon Kinesis inFluent Bit: Manuale ufficiale.

CloudWatch

Creazione di una **`ConfigMap`** per CloudWatch

Sono disponibili due opzioni di output quando si utilizza: CloudWatch

- Un plug-in di output scritto in C
- Un plug-in di output scritto in Golang

L'esempio seguente mostra come utilizzare il `cloudwatch_logs` plugin a cui inviare CloudWatch i log.

1. Salva i contenuti seguenti in un file denominato *`aws-logging-cloudwatch-configmap`*`.yaml`. *`region-code`*Sostituiscilo con Regione AWS quello in cui si trova il tuo cluster. I parametri in `[OUTPUT]` sono obbligatori.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  flb_log_cw: "false"  # Set to true to ship Fluent Bit process logs to
 CloudWatch.
  filters.conf: |
    [FILTER]
        Name parser
        Match *
        Key_name log
        Parser crio
    [FILTER]
        Name kubernetes
        Match kube.*
        Merge_Log On
        Keep_Log Off
        Buffer_Size 0
        Kube_Meta_Cache_TTL 300s
  output.conf: |
    [OUTPUT]
        Name cloudwatch_logs
```

```
        Match   kube.*
        region region-code
        log_group_name my-logs
        log_stream_prefix from-fluent-bit-
        log_retention_days 60
        auto_create_group true
  parsers.conf: |
    [PARSER]
        Name crio
        Format Regex
        Regex ^(?<time>[^ ]+) (?<stream>stdout|stderr) (?<logtag>P|F) (?
<log>.*)$
        Time_Key    time
        Time_Format %Y-%m-%dT%H:%M:%S.%L%z
```

2. Applica il manifesto al cluster.

```
kubectl apply -f aws-logging-cloudwatch-configmap.yaml
```

3. Scarica la policy CloudWatch IAM sul tuo computer. Puoi anche visualizzare la politica su GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-
logging-examples/mainline/examples/fargate/cloudwatchlogs/permissions.json
```

Amazon OpenSearch Service

Per creare un OpenSearch servizio **ConfigMap** per Amazon

Se desideri inviare log ad Amazon OpenSearch Service, puoi utilizzare es output, che è un plug-in scritto inC. L'esempio seguente mostra come utilizzare il plug-in a cui inviare i log. OpenSearch

1. Salva i contenuti seguenti in un file denominato *aws-logging-opensearch-configmap*.yaml. Sostituisci ogni *example value* con i valori in tuo possesso.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
```

```
data:
  output.conf: |
    [OUTPUT]
      Name  es
      Match *
      Host  search-example-gjxdcilagiprbglqn42jsty66y.region-
code.es.amazonaws.com
      Port  443
      Index example
      Type  example_type
      AWS_Auth On
      AWS_Region region-code
      tls   On
```

2. Applica il manifesto al cluster.

```
kubectl apply -f aws-logging-opensearch-configmap.yaml
```

3. Scarica la policy OpenSearch IAM sul tuo computer. Puoi anche visualizzare la politica su GitHub.

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-
fluent-logging-examples/mainline/examples/fargate/amazon-elasticsearch/
permissions.json
```

   Assicurati che il controllo OpenSearch degli accessi di Dashboards sia configurato correttamente. `all_access role`In OpenSearch Dashboards deve essere mappato il ruolo di esecuzione di Pod Fargate e il ruolo IAM. La stessa mappatura deve essere eseguita per il ruolo `security_manager`. È possibile aggiungere i mapping precedenti selezionando `Menu`, `Security`, `Roles`, quindi selezionare i rispettivi ruoli. Per ulteriori informazioni, consulta Come posso risolvere i problemi relativi ai CloudWatch log in modo che vengano trasmessi al mio dominio Amazon ES? .

Firehose

Per creare un file **`ConfigMap`** per Firehose

Sono disponibili due opzioni di output per l'invio dei log a Firehose:

- `kinesis_firehose`: un plug-in di output scritto in C.

- `firehose`: un plug-in di output scritto in Golang.

L'esempio seguente mostra come utilizzare il `kinesis_firehose` plugin per inviare log a Firehose.

1. Salva i contenuti seguenti in un file denominato *`aws-logging-firehose-configmap`*`.yaml`. *`region-code`*Sostituiscilo con quello in Regione AWS cui si trova il tuo cluster.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
data:
  output.conf: |
    [OUTPUT]
     Name  kinesis_firehose
     Match *
     region region-code
     delivery_stream my-stream-firehose
```

2. Applica il manifesto al cluster.

```
kubectl apply -f aws-logging-firehose-configmap.yaml
```

3. Scarica la policy Firehose IAM sul tuo computer. È inoltre possibile visualizzare la politica su. GitHub

```
curl -O https://raw.githubusercontent.com/aws-samples/amazon-eks-fluent-logging-examples/mainline/examples/fargate/kinesis-firehose/permissions.json
```

3. Crea una policy IAM utilizzando il file della policy scaricato nella fase precedente.

```
aws iam create-policy --policy-name eks-fargate-logging-policy --policy-document
 file://permissions.json
```

4. Collega la policy IAM al ruolo di esecuzione del pod specificato per il profilo Fargate con il seguente comando. Sostituisci *`111122223333`* con l'ID del tuo account. Sostituisci

AmazonEKSFargatePodExecutionRole con il ruolo di esecuzione del Pod (per ulteriori informazioni, consulta Creazione di un ruolo di esecuzione del Pod Fargate).

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/eks-fargate-logging-policy \
  --role-name AmazonEKSFargatePodExecutionRole
```

Supporto filtri Kubernetes

Questa funzionalità richiede la versione di Kubernetes minima e il livello di piattaforma riportati di seguito o successivi.

Versione Kubernetes	Livello di piattaforma
1.23 e versioni successive	eks.1

Il filtro Fluent Bit Kubernetes consente di aggiungere metadati Kubernetes ai file di log. Per ulteriori informazioni sul filtro, consulta Kubernetesnella documentazione di Fluent Bit. È possibile applicare un filtro utilizzando l'endpoint del server API.

```
filters.conf: |
    [FILTER]
        Name             kubernetes
        Match            kube.*
        Merge_Log           On
        Buffer_Size         0
        Kube_Meta_Cache_TTL 300s
```

Important

- `Kube_URL`, `Kube_CA_File`, `Kube_Token_Command` e `Kube_Token_File` sono parametri di configurazione di proprietà del servizio e non devono essere specificati. Amazon EKS Fargate popola tali valori.
- `Kube_Meta_Cache_TTL` è il momento in cui Fluent Bit attende di comunicare i metadati più recenti al server API. Se `Kube_Meta_Cache_TTL` non è specificato, Amazon EKS Fargate aggiunge un valore predefinito di 30 minuti per ridurre il carico sul server API.

Invio dei log di processo Fluent Bit all'account

Facoltativamente, puoi spedire i log dei Fluent Bit processi ad Amazon CloudWatch utilizzando quanto segue. `ConfigMap` La spedizione dei log di processo Fluent Bit a CloudWatch richiede costi aggiuntivi di inserimento e archiviazione dei log. Sostituiscilo *`region-code`* con Regione AWS quello in cui si trova il tuo cluster.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: aws-logging
  namespace: aws-observability
  labels:
data:
  # Configuration files: server, input, filters and output
  # ======================================================
  flb_log_cw: "true"  # Ships Fluent Bit process logs to CloudWatch.

  output.conf: |
    [OUTPUT]
        Name cloudwatch
        Match kube.*
        region region-code
        log_group_name fluent-bit-cloudwatch
        log_stream_prefix from-fluent-bit-
        auto_create_group true
```

I log si trovano nell'area in Regione AWS cui risiede il cluster. CloudWatch Il nome del gruppo di registro è *`my-cluster`*`-fluent-bit-logs` e il nome del flusso di log Fluent Bit è `fluent-bit-`*`podname`*`-`*`pod-namespace`*.

> **Note**
>
> - I registri dei processi vengono inviati solo quando il processo Fluent Bit viene avviato correttamente. Se si verifica un errore durante l'avvio di Fluent Bit, i registri di processo non vengono inviati. È possibile inviare i log dei processi solo a. CloudWatch
> - Per eseguire il debug dei registri del processo di invio sull'account, puoi applicare la `ConfigMap` precedente per ottenere i registri del processo. Il mancato avvio di Fluent Bit

solitamente è dovuto al fatto che la `ConfigMap` non viene analizzata o accettata da Fluent Bit durante l'avvio.

Per interrompere i registri dei processi Fluent Bit

I registri Fluent Bit del processo di spedizione CloudWatch richiedono costi aggiuntivi di inserimento e archiviazione dei registri. Per escludere i registri dei processii in una configurazione `ConfigMap` esistente, effettua le seguenti operazioni.

1. Individua il gruppo di CloudWatch log creato automaticamente per i log di Fluent Bit processo del cluster Amazon EKS dopo aver abilitato la registrazione Fargate. Segue il formato `{cluster_name}-fluent-bit-logs`.
2. Elimina i flussi di CloudWatch log esistenti creati per ogni log di Pod's processo nel gruppo di log. CloudWatch
3. Modifica la `ConfigMap` e imposta `flb_log_cw: "false"`.
4. Riavvia tutti i Pods esistenti nel cluster.

Applicazione di prova

1. Implementa un Pod di esempio.

   a. Salva nel tuo computer i seguenti contenuti in un file denominato *`sample-app`*`.yaml`.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: sample-app
  namespace: same-namespace-as-your-fargate-profile
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
```

```
      containers:
        - name: nginx
          image: nginx:latest
          ports:
            - name: http
              containerPort: 80
```

b. Applica il file manifesto al cluster.

```
kubectl apply -f sample-app.yaml
```

2. Visualizza i log NGINX utilizzando la/le destinazione/i configurata/e nel `ConfigMap`.

Considerazioni sulle dimensioni

Si consiglia di pianificare fino a 50 MB di memoria per il router di log. Se prevedi che l'applicazione generi registri a velocità di trasmissione effettiva molto elevata, dovresti pianificarla fino a 100 MB.

Risoluzione dei problemi

Per verificare se la caratteristica di registrazione è abilitata o disabilitata per qualche motivo, ad esempio per una `ConfigMap` non valida, e perché questa non è valida, controlla gli eventi del Pod con **`kubectl describe pod`** ***`pod_name`***. L'output potrebbe includere eventi del Pod che chiariscono se la registrazione è abilitata o meno, ad esempio l'output di esempio seguente.

```
[...]
Annotations:          CapacityProvisioned: 0.25vCPU 0.5GB
                      Logging: LoggingDisabled: LOGGING_CONFIGMAP_NOT_FOUND
                      kubernetes.io/psp: eks.privileged
[...]
Events:
  Type     Reason           Age        From
               Message
  ----     ------           ----       ----
               -------
  Warning  LoggingDisabled  <unknown>  fargate-scheduler
                Disabled logging because aws-logging configmap was not found. configmap
 "aws-logging" not found
```

Gli eventi del Pod sono effimeri, con un periodo di tempo che dipende dalle impostazioni. È inoltre possibile visualizzare le annotazioni di un Pod's utilizzando **`kubectl describe pod`** ***`pod-`***

name. Nell'annotazione del Pod, è possibile verificare se, e per quale motivo, la caratteristica di registrazione è abilitata o disabilitata.

Scelta di un tipo di istanza Amazon EC2

Amazon EC2 offre un'ampia gamma di tipi di istanze per i nodi worker. Ogni tipo di istanza mette a disposizione diverse capacità di calcolo, memoria, archiviazione e rete ed è raggruppata in una famiglia di istanze in base a tali capacità. Per un elenco, consulta i tipi di istanze disponibili nella Amazon EC2 User Guide e i tipi di istanze disponibili nella Amazon EC2 User Guide. Amazon EKS rilascia diverse varianti delle AMI Amazon EC2 per abilitare il supporto. Per assicurarti che il tipo di istanza selezionato sia compatibile con Amazon EKS, prendi in considerazione gli elementi seguenti:

- Al momento, nessuna AMI Amazon EKS supporta le famiglie `g5g` e `mac`.
- Le AMI Amazon EKS Arm e non accelerate non supportano le famiglie `g3`, `g4`, `inf` e `p`.
- Le AMI Amazon EKS accelerate non supportano le famiglie `a`, `c`, `hpc`, `m` e `t`.
- Per le istanze basate su ARM, Amazon Linux 2023 (AL2023) supporta solo tipi di istanze che utilizzano processori Graviton2 o versioni successive. AL2023 non supporta le istanze. `A1`

Quando scegli i tipi di istanza supportati da Amazon EKS, considera le caratteristiche seguenti per ogni tipo.

Numero di istanze in un gruppo di nodi

> In generale, istanze in numero ridotto e di grandi dimensioni sono più convenienti, specialmente se si dispone di molti Daemonsets. Ogni istanza richiede chiamate API al server API, per cui maggiore è il numero di istanze di cui si dispone, maggiore è il carico sul server API.

Sistema operativo

> Esamina i tipi di istanza supportati per Linux, Windows e Bottlerocket. Prima di creare istanze Windows, rivedi Abilitazione del supporto di Windows per il cluster Amazon EKS.

Architettura hardware

> Hai bisogno di x86 o Arm? Puoi implementare Linux solo su Arm. Prima di implementare le istanze Arm, consulta AMI Amazon Linux Arm ottimizzate per Amazon EKS. Ti servono istanze basate sul Nitro System (Linux o Windows) o con un'elaborazione accelerata? Se hai bisogno di funzionalità accelerate, puoi usare Linux solo con Amazon EKS.

Numero massimo di Pods

Dal momento che a ogni Pod viene assegnato il proprio indirizzo IP, il numero di indirizzi IP supportati da un tipo di istanza è un fattore importante per determinare il numero di Pods che possono essere eseguiti sull'istanza. Per determinare manualmente il numero di Pods supportati da un tipo di istanza, consulta Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods.

> **Note**
>
> Con un'AMI Amazon Linux 2 versione `v20220406` o più recente ottimizzata per Amazon EKS, puoi utilizzare un nuovo tipo di istanza senza eseguire l'aggiornamento all'AMI più recente. In questi casi, l'AMI calcola automaticamente il valore `max-pods` necessario se non è elencato nel file `eni-max-pods.txt`. Per impostazione predefinita, i tipi di istanza attualmente in anteprima potrebbero non essere supportati da Amazon EKS. I valori `max-pods` per tali tipi di istanza devono ancora essere aggiunti nel file `eni-max-pods.txt` dell'AMI.

AWS I tipi di istanze Nitro System supportano opzionalmente un numero significativamente maggiore di indirizzi IP rispetto ai tipi di istanze non Nitro System. Tuttavia, non tutti gli indirizzi IP assegnati ad un'istanza sono disponibili per i Pods. Per assegnare un numero significativamente maggiore di indirizzi IP alle istanze, è necessario che la versione `1.9.0` o successiva del componente aggiuntivo CNI di Amazon VPC sia installata nel cluster e che sia configurata in modo appropriato. Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2. Per assegnare il maggior numero di indirizzi IP alle istanze, è necessario che la versione `1.10.1` o successiva del componente aggiuntivo CNI di Amazon VPC sia installata nel cluster e che si implementi il cluster con la famiglia `IPv6`.

Famiglia di IP

Per la creazione di un cluster tramite la famiglia `IPv4`, puoi scegliere qualsiasi tipo di istanza supportata, il che consente al cluster di assegnare indirizzi `IPv4` privati ai tuoi Pods e servizi. Se tuttavia desideri impiegare la famiglia `IPv6` per il cluster, utilizza i tipi di istanza AWS Nitro System o bare metal. Per le istanze Windows è supportato solo `IPv4`. Il cluster deve eseguire la versione `1.10.1` o successiva del componente aggiuntivo CNI di Amazon VPC. Per ulteriori informazioni sull'utilizzo di `IPv6`, consultare IPv6indirizzi per cluster Pods e services.

Versione del componente aggiuntivo CNI di Amazon VPC in uso

La versione più recente del plug-in CNI di Amazon VPC per Kubernetes supporta i seguenti tipi di istanza. Per sfruttare i tipi più recenti di istanza supportati, potrebbe essere necessario aggiornare la versione del componente aggiuntivo CNI di Amazon VPC. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS. L'ultima versione supporta le funzionalità più recenti per l'utilizzo con Amazon EKS. Le versioni precedenti non supportano tutte le funzionalità. Puoi visualizzare le funzionalità supportate dalle diverse versioni in Changelog su GitHub.

Regione AWS in cui stai creando i tuoi nodi

Non tutti i tipi di istanze sono disponibili in tutte le Regioni AWS.

Utilizzo dei gruppi di sicurezza per i Pods

Se si usano gruppi di sicurezza per i Pods, sono supportati solo tipi specifici di istanza. Per ulteriori informazioni, consulta Gruppi di sicurezza per Pods.

Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods

Dal momento che a ogni Pod viene assegnato il proprio indirizzo IP, il numero di indirizzi IP supportati da un tipo di istanza è un fattore importante per determinare il numero di Pods che possono essere eseguiti sull'istanza. Amazon EKS fornisce uno script da scaricare ed eseguire per determinare il numero massimo di Pods consigliato da Amazon EKS per l'esecuzione su ciascun tipo di istanza. Lo script utilizza gli attributi hardware di ogni istanza e le opzioni di configurazione per determinare il numero massimo di Pods. Puoi utilizzare il numero restituito in questi passaggi per abilitare funzionalità quali l'assegnazione di indirizzi IP ai Pods da una sottorete diversa da quella dell'istanza e l'aumento significativo del numero di indirizzi IP dell'istanza. Se utilizzi un gruppo di nodi gestiti con più tipi di istanza, scegli un valore in grado di gestire tutti i tipi di istanza.

1. Esegui il download di uno script che puoi utilizzare per calcolare il numero massimo di Pods per ogni tipo di istanza.

```
curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/templates/
al2/runtime/max-pods-calculator.sh
```

2. Contrassegnare lo script come eseguibile sul computer.

```
chmod +x max-pods-calculator.sh
```

3. Esegui lo script, sostituendo *m5.large* con il tipo di istanza che desideri implementare e *1.9.0-eksbuild.1* con la tua versione del componente aggiuntivo CNI di Amazon VPC. Per determinare la versione del componente aggiuntivo, consultare le procedure di aggiornamento in Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.

   ```
   ./max-pods-calculator.sh --instance-type m5.large --cni-version 1.9.0-eksbuild.1
   ```

   Di seguito viene riportato un output di esempio:

   ```
   29
   ```

   Puoi aggiungere le seguenti opzioni allo script per visualizzare il numero massimo di Pods supportati quando si utilizzano funzionalità opzionali.

   - `--cni-custom-networking-enabled`: utilizzare questa opzione quando si desidera assegnare indirizzi IP da una sottorete diversa da quella della propria istanza. Per ulteriori informazioni, consulta Rete personalizzata per i pod. L'aggiunta di questa opzione allo script precedente, con gli stessi valori di esempio, produce `20`.
   - `--cni-prefix-delegation-enabled`: utilizzare questa opzione quando si desidera assegnare un numero significativo di indirizzi IP a ciascuna interfaccia di rete elastica. Questa funzionalità richiede un'istanza Amazon Linux eseguita su Nitro System e la versione `1.9.0` o successiva del componente aggiuntivo CNI di Amazon VPC. Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2. L'aggiunta di questa opzione allo script precedente, con gli stessi valori di esempio, produce `110`.

È inoltre possibile eseguire lo script con l'opzione `--help` per visualizzare tutte le opzioni disponibili.

> **Note**
>
> Lo script del calcolatore di Pods massimi limita il valore restituito a `110` in base alle soglie di scalabilità di Kubernetes e alle impostazioni consigliate. Se il tipo di istanza ha più di 30 vCPU, questo limite sale a `250`, un numero basato sui test interni del team di scalabilità di Amazon EKS. Per ulteriori informazioni, consulta il post del blog dedicato all'aumento dei limiti dei pod per nodo del plug-in CNI di Amazon VPC.

AMI ottimizzate per Amazon EKS

È possibile implementare i nodi con versioni preconfigurate delle Amazon Machine Image (AMI) ottimizzate per Amazon EKS o con AMI personalizzate. Per ulteriori informazioni sui vari tipi di AMI ottimizzate per Amazon EKS, consulta il relativo argomento di seguito. Per istruzioni su come creare un'AMI personalizzata, consulta Script di build per AMI Amazon Linux ottimizzata per Amazon EKS.

Argomenti

Amazon EKS ha terminato il supporto per **`Dockershim`**

Kubernetes non supporta più `Dockershim`. Il team di Kubernetes ha rimosso completamente il runtime nella versione `1.24` di Kubernetes. Per ulteriori informazioni, consulta Kubernetes si lascia alle spalle Dockershim: impegni e passaggi successivi sul Blog di Kubernetes.

Amazon EKS ha interrotto il supporto anche per `Dockershim` a partire dal rilascio della versione `1.24` di Kubernetes. Le AMI Amazon EKS pubblicate ufficialmente includono `containerd` come unico runtime a partire dalla versione `1.24`. Questo argomento contiene alcuni dettagli, ma ulteriori informazioni sono disponibili in Tutto quello che devi sapere sul passaggio a containerd su Amazon EKS.

Esiste un plug-in `kubectl` che puoi utilizzare per visualizzare quali carichi di lavoro Kubernetes montano il volume del socket Docker. Per ulteriori informazioni, consulta Detector for Docker Socket (DDS) su GitHub. Le AMI Amazon EKS che eseguono versioni di Kubernetes precedenti a `1.24` utilizzano Docker come runtime predefinito. Tuttavia, queste AMI EKS Amazon hanno un'opzione flag di bootstrap che consente di testare i carichi di lavoro su qualunque cluster supportato con `containerd`. Per ulteriori informazioni, consulta Prova la migrazione da Docker a `containerd`.

Continueremo a pubblicare le AMI per le versioni esistenti di Kubernetes fino alla fine della data di supporto. Per ulteriori informazioni, consulta Calendario dei rilasci Kubernetes Amazon EKS. Se occorre più tempo per testare i carichi di lavoro in `containerd`, utilizza una versione supportata

anteriore a `1.24`. Tuttavia, per aggiornare le AMI Amazon EKS ufficiali alla versione `1.24` o successiva, accertati che i tuoi carichi di lavoro funzionino su `containerd`.

Il runtime `containerd` offre prestazioni e sicurezza più affidabili. `containerd` è il runtime standardizzato in Amazon EKS. Fargate e Bottlerocket già utilizzano solo `containerd`. `containerd` consente di ridurre al minimo il numero di versioni dell'AMI Amazon EKS necessarie per gestire CVE (Common Vulnerabilities and Exposures) di `Dockershim`. Dal momento che `Dockershim` già utilizza `containerd` internamente, potrebbe non essere necessario apportare modifiche. che sono richieste, invece, in alcune situazioni:

- Devi apportare modifiche alle applicazioni che montano il socket Docker. Ad esempio, sono interessate le immagini di container generate con un container. Anche numerosi strumenti di monitoraggio montano il socket Docker. Potrebbe essere necessario attendere aggiornamenti o implementare nuovamente i carichi di lavoro per il monitoraggio del runtime.
- Potrebbe essere necessario apportare modifiche per le applicazioni che si basano su impostazioni Docker specifiche. Ad esempio, il protocollo `HTTPS_PROXY` non è più supportato. Devi aggiornare le applicazioni che utilizzano questo protocollo. Per ulteriori informazioni, consulta `dockerd` nella Documentazione di Docker.
- Se utilizzi l'assistente credenziali Amazon ECR per estrarre immagini, devi passare al provider di credenziali per immagini `kubelet`. Per ulteriori informazioni, consulta Configurare un provider di credenziali per immagini `kubelet` nella documentazione di Kubernetes.
- Siccome Amazon EKS `1.24` non supporta più Docker, alcuni flag che lo Script di bootstrap di Amazon EKS supportava precedentemente non sono più supportati. Prima di passare ad Amazon EKS `1.24` o versioni successive, devi rimuovere tutti i riferimenti ai flag che attualmente non sono supportati:
  - `--container-runtime dockerd` (`containerd` è l'unico valore supportato)
  - `--enable-docker-bridge`
  - `--docker-config-json`
- Se hai già configurato Fluentd per Container Insights, devi eseguire la migrazione di Fluentd a Fluent Bit prima di passare a `containerd`. I parser Fluentd sono configurati per analizzare solo i messaggi di log in formato JSON. A differenza di `dockerd`, il runtime del container `containerd` contiene messaggi di log che non sono in formato JSON. Se non esegui la migrazione a Fluent Bit, alcuni dei parser Fluentd's configurati genereranno un'enorme quantità di errori all'interno del container Fluentd. Per ulteriori informazioni sulla migrazione, consulta Configurare Fluent Bit come invio DaemonSet di log a Logs. CloudWatch

- Se utilizzi un'AMI personalizzata e stai effettuando l'upgrade ad `1.24` Amazon EKS, devi assicurarti che l'inoltro IP sia abilitato per i tuoi nodi worker. Questa impostazione non era necessaria con Docker ma è necessaria per `containerd`. È necessario per risolvere i problemi di connettività di rete Pod-to-Pod, da Pod verso l'esterno oppure Pod-to-apiserver.

  Per verificare questa impostazione su un nodo worker, esegui uno dei seguenti comandi:

  - `sysctl net.ipv4.ip_forward`
  - `cat /proc/sys/net/ipv4/ip_forward`

  Se l'output è `0`, esegui uno dei seguenti comandi per attivare la variabile kernel `net.ipv4.ip_forward`:

  - `sysctl -w net.ipv4.ip_forward=1`
  - `echo 1 > /proc/sys/net/ipv4/ip_forward`

  Per l'attivazione dell'impostazione sulle AMI di Amazon EKS in fase di runtime `containerd`, consulta `install-worker.sh` su GitHub.

AMI Amazon Linux ottimizzate per Amazon EKS

L'AMI Amazon Linux ottimizzata per Amazon EKS è basata su Amazon Linux 2 (AL2) e Amazon Linux 2023 (AL2023). È configurata per fungere da immagine di base per i nodi Amazon EKS. L'AMI è configurata per l'uso con Amazon EKS e include i seguenti componenti:

- `kubelet`
- AWS Autenticatore IAM
- Docker (Amazon EKS versione `1.23` e precedenti)
- `containerd`

> **Note**
>
> - Puoi tenere traccia degli eventi di sicurezza o privacy per AL2 presso il centro di sicurezza di Amazon Linux o abbonarti al feed RSS associato. Gli eventi di sicurezza e privacy includono una panoramica del problema, quali sono i pacchetti interessati e come aggiornare le istanze per risolvere il problema.

- Prima di implementare un'AMI accelerata o di tipo Arm, consulta le informazioni in AMI Amazon Linux accelerata ottimizzata per Amazon EKS e AMI Amazon Linux Arm ottimizzate per Amazon EKS.
- Per quanto riguarda la Kubernetes versione`1.23`, puoi utilizzare un flag bootstrap opzionale per testare la migrazione da a. Docker `containerd` Per ulteriori informazioni, consulta Prova la migrazione da Docker a `containerd`.
- A partire da Kubernetes versione `1.25`, non sarai più in grado di utilizzare le istanze `P2` di Amazon EC2 con le AMI Amazon Linux accelerate ottimizzate e preconfigurate per Amazon EKS. Queste AMI per versioni di Kubernetes `1.25` o successive supporteranno driver di serie `NVIDIA 525` o successive, che non sono compatibili con le istanze `P2`. Tuttavia, i driver della serie `NVIDIA 525` o versioni successive sono compatibili con le istanze `P3`, `P4` e `P5`, quindi puoi utilizzare quelle istanze con le AMI per Kubernetes versione `1.25`. Prima che i cluster Amazon EKS vengano aggiornati alla versione `1.25`, migrano qualsiasi istanza `P2` su istanze `P3`, `P4`, e `P5`. È inoltre necessario aggiornare in modo proattivo le applicazioni per funzionare con la serie `NVIDIA 525` o successive. Abbiamo intenzione di effettuare il backport dei driver della `NVIDIA 525` serie più recente o successiva alle Kubernetes versioni `1.23` e alla `1.24` fine di gennaio 2024.
- Tutti i gruppi di nodi gestiti appena creati nei cluster alla versione `1.30` o successiva utilizzeranno automaticamente AL2023 come sistema operativo del nodo. In precedenza, i nuovi gruppi di nodi utilizzavano per impostazione predefinita AL2. È possibile continuare a utilizzare AL2 scegliendolo come tipo di AMI durante la creazione di un nuovo gruppo di nodi.
- Il supporto per AL2 terminerà il 30 giugno 2025. Per ulteriori informazioni, consulta Domande frequenti su Amazon Linux 2.

Aggiornamento da AL2 a AL2023

L'AMI ottimizzata per Amazon EKS è disponibile in due famiglie basate su AL2 e AL2023. AL2023 è un nuovo sistema operativo basato su Linux progettato per fornire un ambiente sicuro, stabile e ad alte prestazioni per le tue applicazioni cloud. È la nuova generazione di Amazon Linux di Amazon Web Services ed è disponibile in tutte le versioni supportate di Amazon EKS, incluse le versioni `1.23` e `1.24` con supporto esteso. Le AMI accelerate di Amazon EKS basate su AL2023 saranno disponibili in un secondo momento. Se hai carichi di lavoro accelerati, dovresti continuare a utilizzare l'AMI accelerata AL2 o Bottlerocket.

AL2023 offre diversi miglioramenti rispetto ad AL2. Per un confronto completo, consulta Comparing AL2 e Amazon Linux 2023 nella Amazon Linux 2023 User Guide. Diversi pacchetti sono stati aggiunti, aggiornati e rimossi da AL2. Si consiglia vivamente di testare le applicazioni con AL2023 prima dell'aggiornamento. Per un elenco di tutte le modifiche ai pacchetti in AL2023, consulta Modifiche ai pacchetti in Amazon Linux 2023 nelle Note di rilascio di Amazon Linux 2023.

Oltre a queste modifiche, dovresti essere consapevole di quanto segue:

- AL2023 introduce un nuovo processo di inizializzazione dei nodi `nodeadm` che utilizza uno schema di configurazione YAML. Se utilizzi gruppi di nodi autogestiti o un'AMI con un modello di avvio, ora dovrai fornire esplicitamente metadati del cluster aggiuntivi quando crei un nuovo gruppo di nodi. Un esempio dei parametri minimi richiesti è il seguente `apiServerEndpointcertificateAuthority`, dove ora `cidr` sono richiesti e il servizio:

  ```
  ---
  apiVersion: node.eks.aws/v1alpha1
  kind: NodeConfig
  spec:
    cluster:
      name: my-cluster
      apiServerEndpoint: https://example.com
      certificateAuthority: Y2VydGlmaWNhdGVBdXRob3JpdHk=
      cidr: 10.100.0.0/16
  ```

  In AL2, i metadati di questi parametri sono stati rilevati dalla chiamata `DescribeCluster` API di Amazon EKS. Con AL2023, questo comportamento è cambiato poiché la chiamata API aggiuntiva rischia di rallentare durante l'up-up di nodi su larga scala. Questa modifica non ha effetto su di te se utilizzi gruppi di nodi gestiti senza un modello di avvio o se utilizzi. Karpenter Per ulteriori informazioni sul `certificateAuthority` servizio`cidr`, consulta `DescribeCluster` la pagina Amazon EKS API Reference.
- Dockernon è supportato in AL2023 per tutte le versioni di Amazon EKS supportate. Il supporto per Docker è terminato ed è stato rimosso con la versione Amazon EKS `1.24` o successiva in AL2. Per ulteriori informazioni sull'obsolescenza, consulta Amazon EKS ha terminato il supporto per. `Dockershim`
- Per AL2023 è richiesta la versione Amazon VPC CNI `1.16.2` o superiore.
- AL2023 richiede per impostazione predefinita. `IMDSv2` `IMDSv2`presenta diversi vantaggi che aiutano a migliorare il livello di sicurezza. Utilizza un metodo di autenticazione orientato alla sessione che richiede la creazione di un token segreto in una semplice richiesta HTTP PUT per

avviare la sessione. Il token di una sessione può essere valido per un periodo compreso tra 1 secondo e 6 ore. Per ulteriori informazioni su come passare da a`IMDSv2`, consulta Transizione `IMDSv1` all'utilizzo di Instance Metadata Service versione 2 e Sfrutta tutti i vantaggi di IMDSv2 e disabilita IMDSv1 nell'infrastruttura. AWS Se desideri utilizzarlo`IMDSv1`, puoi comunque farlo sovrascrivendo manualmente le impostazioni utilizzando le proprietà di avvio dell'opzione dei metadati dell'istanza.

> **Note**
>
> Infatti`IMDSv2`, il numero di hop predefinito per i gruppi di nodi gestiti è impostato su 1. Ciò significa che i contenitori non avranno accesso alle credenziali del nodo tramite IMDS. Se hai bisogno dell'accesso del contenitore alle credenziali del nodo, puoi comunque farlo sovrascrivendo manualmente le credenziali `HttpPutResponseHopLimit` in un modello di lancio Amazon EC2 personalizzato, aumentandolo a 2. In alternativa, puoi utilizzare Amazon EKS Pod Identity per fornire credenziali anzichéIMDSv2.

- AL2023 presenta la nuova generazione di gerarchia unificata dei gruppi di controllo (). `cgroupv2` `cgroupv2`viene utilizzato per implementare un runtime di container e da. `systemd` Sebbene AL2023 includa ancora codice che può far funzionare il sistema`cgroupv1`, questa non è una configurazione consigliata o supportata. Questa configurazione verrà completamente rimossa in una delle future release principali di Amazon Linux.
- `eksctl`è richiesta una versione `0.176.0` o superiore per `eksctl` supportare AL2023.

Per i gruppi di nodi gestiti in precedenza, puoi eseguire un aggiornamento sul posto o un aggiornamento blu/verde a seconda di come stai utilizzando un modello di lancio:

- Se utilizzi un'AMI personalizzata con un gruppo di nodi gestito, puoi eseguire un aggiornamento sul posto scambiando l'ID AMI nel modello di avvio. È necessario assicurarsi che le applicazioni e tutti i dati utente vengano trasferiti ad AL2023 prima di eseguire questa strategia di aggiornamento.
- Se utilizzi gruppi di nodi gestiti con il modello di avvio standard o con un modello di avvio personalizzato che non specifica l'ID AMI, devi eseguire l'aggiornamento utilizzando una strategia blu/verde. Un aggiornamento blu/verde è in genere più complesso e comporta la creazione di un gruppo di nodi completamente nuovo in cui specificare AL2023 come tipo AMI. Il nuovo gruppo di nodi dovrà quindi essere configurato con cura per garantire che tutti i dati personalizzati del gruppo di nodi AL2 siano compatibili con il nuovo sistema operativo. Una volta che il nuovo gruppo di nodi è stato testato e convalidato con le applicazioni, Pods può essere migrato dal vecchio gruppo di

nodi al nuovo gruppo di nodi. Una volta completata la migrazione, puoi eliminare il vecchio gruppo di nodi.

Se utilizzi Karpenter e desideri utilizzare AL2023, dovrai modificare il `EC2NodeClass` `amiFamily` campo con AL2023. Per impostazione predefinita, Drift è abilitato in. Karpenter Ciò significa che, una volta modificato il `amiFamily` campo, Karpenter aggiornerà automaticamente i nodi di lavoro all'AMI più recente, quando disponibile.

AMI Amazon Linux accelerata ottimizzata per Amazon EKS

> **Note**
>
> Le AMI accelerate di Amazon EKS basate su AL2023 saranno disponibili in un secondo momento. Se disponi di carichi di lavoro accelerati, dovresti continuare a utilizzare l'AMI accelerata AL2 o. Bottlerocket

L'AMI Amazon Linux accelerata ottimizzata per Amazon EKS è basata sull'AMI Amazon Linux standard ottimizzata per Amazon EKS. È configurato per fungere da immagine opzionale per i nodi Amazon EKS per supportare carichi di lavoro basati su GPU, Inferentia e Trainium.

Oltre alla configurazione standard dell'AMI ottimizzata per Amazon EKS, l'AMI accelerata include quanto segue:

- Driver NVIDIA
- `nvidia-container-runtime`
- AWS Neuronautista

Per un elenco dei componenti più recenti inclusi nell'AMI accelerata, consulta le `amazon-eks-ami` versioni suGitHub.

> **Note**
>
> - L'AMI accelerata ottimizzata per Amazon EKS supporta solo i tipi di istanza basati su GPU e Inferentia. Assicurati di specificare questi tipi di istanza nel tuo AWS CloudFormation modello di nodo. Utilizzando l'AMI accelerata ottimizzata per Amazon EKS, si accettano i termini del contratto di licenza con l'utente finale (EULA) di NVIDIA.

- L'AMI accelerata ottimizzata per Amazon EKS era precedentemente denominata AMI ottimizzata per Amazon EKS con supporto GPU.
- Le versioni precedenti dell'AMI accelerata ottimizzata per Amazon EKS prevedevano l'installazione del repository `nvidia-docker`. Il repository non è più incluso nella versione dell'AMI di Amazon EKS `v20200529` e successive.

Per abilitare i carichi di AWS lavoro basati su Neuron (acceleratore ML)

Per i dettagli sui carichi di lavoro di formazione e inferenza utilizzati in Neuron Amazon EKS, consulta i seguenti riferimenti:

- Containers - Kubernetes - Guida introduttiva nella documentazione AWS Neuron
- Formazione su EKS Samples AWS Neuron su GitHub
- Inferenza del machine learning utilizzando AWS Inferentia

Abilitazione dei carichi di lavoro basati su GPU

Nella sezione seguente viene descritto come eseguire un carico di lavoro su un'istanza basata su GPU con l'AMI accelerata e ottimizzata per Amazon EKS.

1. Quando i nodi di lavoro GPU si uniscono al cluster, applicare il plugin del dispositivo NVIDIA per Kubernetes come DaemonSet sul cluster. Sostituisci *vX.X.X* con la versione plugin per dispositivi NVIDIA/K8Sdesiderata. non ospita questa immagine, quindi anche l'organizzazione deve caricare l'immagine.

   ```
   kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/
   nvidia-device-plugin.yml
   ```

2. È possibile verificare che i nodi abbiano GPU allocabili con il seguente comando.

   ```
   kubectl get nodes "-o=custom-
   columns=NAME:.metadata.name,GPU:.status.allocatable.nvidia\.com/gpu"
   ```

Implementazione di un Pod utile a verificare che i nodi della GPU siano configurati correttamente

1. Crea un file denominato `nvidia-smi.yaml` con i seguenti contenuti. Sostituisci *tag* con il tag desiderato per nvidia/cuda. Questo manifesto avvia un container NVIDIA CUDA che esegue `nvidia-smi` su un nodo.

```
apiVersion: v1
kind: Pod
metadata:
  name: nvidia-smi
spec:
  restartPolicy: OnFailure
  containers:
  - name: nvidia-smi
    image: nvidia/cuda:tag
    args:
    - "nvidia-smi"
    resources:
      limits:
        nvidia.com/gpu: 1
```

2. Applicare il file manifesto con il comando seguente.

```
kubectl apply -f nvidia-smi.yaml
```

3. Quando il Pod ha terminato l'esecuzione, visualizzane i registri tramite il comando seguente.

```
kubectl logs nvidia-smi
```

Di seguito viene riportato un output di esempio:

```
Mon Aug  6 20:23:31 20XX
+-----------------------------------------------------------------------------+
| NVIDIA-SMI XXX.XX                 Driver Version: XXX.XX                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  On   | 00000000:00:1C.0 Off |                    0 |
| N/A   46C    P0    47W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
```

```
+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

AMI Amazon Linux Arm ottimizzate per Amazon EKS

Le istanze Arm offrono significativi risparmi sui costi per applicazioni aumentabili orizzontalmente e basate su Arm come server Web, microservizi containerizzati, parchi istanze di memorizzazione nella cache e archivi di dati distribuiti. Quando si aggiungono nodi Arm al cluster, esaminare le considerazioni riportate di seguito.

Considerazioni

- Se il cluster è stato implementato prima del 17 agosto 2020, è necessario eseguire un aggiornamento una tantum dei manifesti critici dei componenti aggiuntivi del cluster. In questo modo Kubernetes può estrarre l'immagine corretta per ogni architettura hardware utilizzata nel cluster. Per ulteriori informazioni sull'aggiornamento dei componenti aggiuntivi del cluster, consultare Aggiornamento della versione di Kubernetes per il cluster Amazon EKS. Se il cluster è stato implementato il 17 agosto 2020 o in una data successiva, i componenti aggiuntivi CoreDNS, `kube-proxy` e Amazon VPC CNI plugin for Kubernetes sono già predisposti per un sistema multi-architettura.
- Le applicazioni implementate nei nodi Arm devono essere compilate per Arm.
- Se si dispone di DaemonSets implementati in un cluster esistente o si desidera distribuirli in un nuovo cluster in cui implementare anche i nodi Arm, verificare che DaemonSet possa essere eseguito su tutte le architetture hardware del cluster.
- È possibile eseguire gruppi di nodi Arm e gruppi di nodi x86 nello stesso cluster. In tal caso, è consigliabile implementare le immagini di container multiarchitettura in un repository del container, come Amazon Elastic Container Registry (Amazon ECR), quindi aggiungere i selettori di nodi ai manifesti in modo che Kubernetes sappia in quale architettura hardware può essere implementato un Pod. Per ulteriori informazioni, consultare Inviare un'immagine multi-architettura nella Guida per l'utente di Amazon ECR ed il blog post Presentazione di immagini container multi-architettura per Amazon ECR.

Prova la migrazione da Docker a `containerd`

Amazon EKS ha interrotto il supporto per Docker a partire dal lancio di Kubernetes versione `1.24`. Per ulteriori informazioni, consulta Amazon EKS ha terminato il supporto per `Dockershim`.

Per quanto riguarda la Kubernetes versione`1.23`, puoi utilizzare un flag bootstrap opzionale per abilitare il `containerd` runtime per le AMI AL2 ottimizzate per Amazon EKS. Questa funzione fornisce un percorso chiaro per la migrazione a `containerd` quando viene eseguito l'aggiornamento alla versione `1.24` o successiva. Amazon EKS ha interrotto il supporto per Docker a partire dal lancio di Kubernetes versione `1.24`. Il runtime di `containerd` è stato ampiamente adottato nella community Kubernetes ed è un progetto promosso con il CNCF (Cloud Native Computer Foundation). È possibile testarlo aggiungendo un gruppo di nodi a un cluster nuovo o esistente.

Creando uno tra i seguenti tipi di gruppi di nodi, è possibile abilitare il flag di bootstrap.

Autogestito

Crea il gruppo di nodi utilizzando le istruzioni riportate in Avvio di nodi Amazon Linux autogestiti. Specifica un'AMI ottimizzata per Amazon EKS e il testo seguente per il parametro `BootstrapArguments`.

```
--container-runtime containerd
```

Gestiti

Se si utilizza `eksctl`, è necessario creare un file denominato *`my-nodegroup`*`.yaml` con i seguenti contenuti. Sostituisci ogni *`example value`* con i valori in tuo possesso. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura. Per recuperare l'ID di un'AMI ottimizzata per `ami-`*`1234567890abcdef0`*, consulta la sezione Recupero ID delle AMI Amazon Linux ottimizzate per Amazon EKS.

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code
  version: 1.23
managedNodeGroups:
  - name: my-nodegroup
```

```
    ami: ami-1234567890abcdef0
    overrideBootstrapCommand: |
      #!/bin/bash
      /etc/eks/bootstrap.sh my-cluster --container-runtime containerd
```

> **Note**
>
> Se si avviano più nodi contemporaneamente, potrebbe essere necessario specificare anche i valori per gli argomenti di bootstrap `--apiserver-endpoint`, `--b64-cluster-ca` e `--dns-cluster-ip`, al fine di evitare errori. Per ulteriori informazioni, consulta Specifica di un'AMI.

Esegui i seguenti comandi per creare il gruppo di nodi.

```
eksctl create nodegroup -f my-nodegroup.yaml
```

Se si preferisce utilizzare uno strumento diverso per creare il gruppo di nodi gestiti, è necessario implementare il gruppo di nodi utilizzando un modello di avvio. Nel modello di avvio, specifica l'ID di un'AMI ottimizzata per Amazon EKS, quindi implementa il gruppo di nodi utilizzando un modello di avvio e fornisci i seguenti dati utente. Questi dati utente passano gli argomenti nel file `bootstrap.sh`. Per ulteriori informazioni sul file bootstrap, consulta bootstrap.sh su GitHub.

```
/etc/eks/bootstrap.sh my-cluster --container-runtime containerd
```

Ulteriori informazioni

Per ulteriori informazioni sull'utilizzo delle AMI Amazon Linux ottimizzate per Amazon EKS, consulta le sezioni seguenti:

- Per utilizzare Amazon Linux con gruppi di nodi gestiti, consulta la sezione Gruppi di nodi gestiti.
- Per avviare nodi Amazon Linux autogestiti, consulta la sezione Recupero ID delle AMI Amazon Linux ottimizzate per Amazon EKS.
- Per informazioni sulla versione, consulta AMI Amazon Linux ottimizzata per Amazon EKS.
- Per recuperare gli ID più recenti delle AMI Amazon Linux ottimizzate per Amazon EKS, consulta la sezione Recupero ID delle AMI Amazon Linux ottimizzate per Amazon EKS.

- Per consultare gli script open-source che vengono utilizzati per creare l'AMI ottimizzata per Amazon EKS, consulta la sezione Script di build per AMI Amazon Linux ottimizzata per Amazon EKS.

AMI Amazon Linux ottimizzata per Amazon EKS

Le versioni delle AMI Amazon Linux ottimizzate per Amazon EKS si basano sulla versione di Kubernetes e la data di rilascio dell'AMI è nel formato seguente:

```
k8s_major_version.k8s_minor_version.k8s_patch_version-release_date
```

Ogni versione di AMI include varie versioni di `kubelet`, Docker, del kernel Linux e di `containerd`. L'AMI accelerata include anche varie versioni del driver NVIDIA. Puoi trovare queste informazioni sulle versioni in Changelog su GitHub.

Recupero ID delle AMI Amazon Linux ottimizzate per Amazon EKS

Puoi recuperare a livello di codice l'ID Amazon Machine Image (AMI) per le AMI ottimizzate per Amazon EKS interrogando l'API Parameter Store. AWS Systems Manager Questo parametro consente di evitare la ricerca manuale degli ID AMI ottimizzata per Amazon EKS. Per ulteriori informazioni sull'API Systems Manager Parameter Store, vedere GetParameter.

Per recuperare un ID AMI per le AMI ottimizzate di Amazon EKS utilizzando AWS CLI

1. Determina la regione in cui verrà distribuita l'istanza del nodo, ad esempio. `us-west-2`
2. Determina il tipo di AMI di cui hai bisogno. Per informazioni sui tipi di istanze Amazon EC2, consulta Tipi di istanze.
   - `amazon-linux-2`è per istanze x86 basate su Amazon Linux 2 (AL2).
   - `amazon-linux-2-arm64`è per istanze ARM AL2, come AWS le istanze basate su Graviton.
   - `amazon-linux-2-gpu`è per istanze accelerate con GPU AL2.
   - `amazon-linux-2023/x86_64/standard`è per istanze x86 basate su Amazon Linux 2023 (AL2023).
   - `amazon-linux-2023/arm64/standard`è per le istanze ARM AL2023.
3. Determina la Kubernetes versione del cluster a cui verrà collegato il nodo, ad esempio 1.30.
4. Esegui il AWS CLI comando seguente per recuperare l'ID AMI appropriato. Sostituisci Regione AWS la Kubernetes versione e la piattaforma appropriate. È necessario accedere AWS CLI

utilizzando un principale IAM con l'autorizzazione IAM per recuperare i `ssm:GetParameter` metadati AMI ottimizzati per Amazon EKS.

```
aws ssm get-parameter --name /aws/service/eks/optimized-ami/1.30/amazon-linux-2/
recommended/image_id \
                --region region-code --query "Parameter.Value" --output text
```

Di seguito viene riportato un output di esempio:

```
ami-1234567890abcdef0
```

Script di build per AMI Amazon Linux ottimizzata per Amazon EKS

Amazon Elastic Kubernetes Service (Amazon EKS) dispone di script open-source che vengono utilizzati per creare l'AMI ottimizzata per Amazon EKS. Questi script della build sono disponibili su GitHub.

L'AMI Amazon Linux ottimizzata per Amazon EKS è basata su Amazon Linux 2 (AL2) e Amazon Linux 2023 (AL2023), specificamente per l'uso come nodo nei cluster Amazon EKS. Puoi utilizzare questo repository per visualizzare le specifiche di configurazione del team Amazon EKS `kubelet`Docker, AWS IAM Authenticator per Kubernetes e creare da zero la tua AMI basata su Amazon Linux.

Il repository di build scripts include un modello di HashiCorppacker e script di compilazione per generare un AMI. Questi script sono la "source of truth" per build AMI ottimizzate per Amazon EKS, perciò puoi seguire il repository GitHub per monitorare le modifiche alle AMI. Ad esempio, magari vuoi che l'AMI utilizzi la stessa versione di Docker utilizzata dal team Amazon EKS per l'AMI ufficiale.

Il GitHub repository contiene anche lo script bootstrap specializzato e lo script nodeadm che vengono eseguiti all'avvio per configurare i dati del certificato dell'istanza, l'endpoint del piano di controllo, il nome del cluster e altro ancora.

Inoltre, il GitHub repository contiene i nostri AWS CloudFormation modelli di nodi Amazon EKS. Questi modelli consentono di rendere operativa un'istanza che esegue l'AMI ottimizzata per Amazon EKS e registrarla con un cluster.

Per ulteriori informazioni, consulta i repository su GitHub all'indirizzo https://github.com/awslabs/amazon-eks-ami.

AL2 ottimizzato per Amazon EKS contiene un flag bootstrap opzionale per abilitare il `containerd` runtime.

Configurazione VT1 per la tua AMI Amazon Linux personalizzata

Le AMI Amazon Linux personalizzate in Amazon EKS possono supportare la famiglia di istanze di transcodifica video VT1 per Amazon Linux 2 (AL2), 18 e 20. Ubuntu Ubuntu VT1supporta le schede di transcodifica multimediale Xilinx U30 con codec H.264/AVC e H.265/HEVC accelerati. Per ottenere il vantaggio di queste istanze accelerate, attenersi alla seguente procedura:

1. Crea e avvia un'AMI di base da AL2, Ubuntu 18 o Ubuntu 20.
2. Dopo l'avvio dell'AMI basato, installare il Driver XRT e tempo di esecuzione sul nodo.
3. Creazione di un cluster Amazon EKS.
4. Installa il plug-in FPGA di Kubernetes nel cluster.

```
kubectl apply -f fpga-device-plugin.yml
```

Il plug-in ora pubblicizzerà i dispositivi Xilinx U30 per nodo sul tuo cluster Amazon EKS. Puoi utilizzare l'immagine FFMPEG docker per eseguire esempi di carichi di lavoro di transcodifica video sul tuo cluster Amazon EKS.

Configurazione DL1 per la tua AMI Amazon Linux 2 personalizzata

Le AMI Amazon Linux 2 (AL2) personalizzate in Amazon EKS possono supportare carichi di lavoro di deep learning su larga scala tramite configurazioni e componenti aggiuntivi. Kubernetes Questo documento descrive i componenti necessari per impostare una soluzione Kubernetes generica per una configurazione on-premise o come punto di partenza per una configurazione cloud più ampia. Per supportare questa funzione, dovrai eseguire le seguenti operazioni nel tuo ambiente personalizzato:

- SynapaseAI® Softwaredriver caricati sul sistema: sono inclusi nelle AMI disponibili su Github.
- Il plug-in del Habana dispositivo: un plug-in DaemonSet che consente di abilitare automaticamente la registrazione dei Habana dispositivi nel Kubernetes cluster e di monitorare lo stato del dispositivo.
- Helm 3.x
- Grafico Helm per installare l'operatore MPI.
- Operatore MPI

1. Crea e avvia un'AMI di base da AL2, Ubuntu 18 o Ubuntu 20.
2. Segui queste istruzioni per configurare l'ambiente perDL1.

AMI Bottlerocket ottimizzate per Amazon EKS

Bottlerocket è una distribuzione Linux open source sponsorizzata e supportata da AWS. Bottlerocket è stato progettato appositamente per ospitare carichi di lavoro in container. Con Bottlerocket puoi migliorare la disponibilità delle implementazioni containerizzate e ridurre i costi operativi automatizzando gli aggiornamenti dell'infrastruttura dei container. Bottlerocket include solo il software essenziale per eseguire i container e ciò migliora l'utilizzo delle risorse, riduce le minacce alla sicurezza e limita il sovraccarico di gestione. L'AMI Bottlerocket include `containerd`, `kubelet`, e AWS IAM Authenticator. Oltre che dai gruppi di nodi gestiti e di nodi autogestiti, Bottlerocket è supportato anche da Karpenter.

Vantaggi

L'utilizzo di Bottlerocket con il cluster Amazon EKS presenta i seguenti vantaggi:

- Maggiore operatività con costi operativi inferiori e minore complessità di gestione: Bottlerocket necessita di un minore utilizzo di risorse, tempi di avvio più brevi ed è meno vulnerabile alle minacce alla sicurezza rispetto ad altre distribuzioni Linux. Un minore utilizzo di risorse Bottlerocket's aiuta a ridurre i costi utilizzando meno risorse di archiviazione, elaborazione e rete.
- Maggiore sicurezza grazie agli aggiornamenti automatici del sistema operativo: gli aggiornamenti di Bottlerocket vengono applicati come una singola unità che può essere ripristinata, se necessario. Ciò elimina il rischio di aggiornamenti danneggiati o non riusciti che possono lasciare il sistema in uno stato inutilizzabile. Con Bottlerocket, gli aggiornamenti di sicurezza possono essere applicati automaticamente non appena sono disponibili con un impatto minimo e possono essere ripristinati in caso di guasti.
- Supporto premium: le build fornite da AWS di Bottlerocket su Amazon EC2 sono coperte dagli stessi piani di AWS Support che coprono anche servizi AWS come Amazon EC2, Amazon EKS e Amazon ECR.

Considerazioni

Quando utilizzi Bottlerocket per il tuo tipo di AMI, considera quanto segue:

- Bottlerocket supporta le istanze Amazon EC2 con processori `x86_64` e `arm64`. L'AMI Bottlerocket non è consigliata per l'uso con istanze Amazon EC2 dotate di chip Inferentia.
- Al momento, non esiste un modello CloudFormation AWS con cui implementare i nodi Bottlerocket.
- Le immagini Bottlerocket non includono un server SSH o uno shell (interprete di comandi). Puoi utilizzare metodi di accesso fuori banda per consentire SSH. Questi approcci consentono di utilizzare il container dell'amministratore e saltare alcune fasi di configurazione di bootstrap con i dati utente. Per ulteriori informazioni, fai riferimento alle sezioni seguenti in OS di Bottlerocket su GitHub:
  - Esplorazione
  - Container amministratore
  - Impostazioni di Kubernetes
- Bottlerocket utilizza diversi tipi di container:
  - Per impostazione predefinita, è abilitato un container di controllo. Questo container gestisce l'agente AWS Systems Manager che è possibile utilizzare per eseguire comandi o avviare sessioni di shell sulle istanze Bottlerocket di Amazon EC2. Per ulteriori informazioni, consulta Configurazione di Session Manager nella Guida per l'utente di AWS Systems Manager.
  - Se viene fornita una chiave SSH durante la creazione del gruppo di nodi, viene abilitato un container dell'amministratore. Consigliamo di utilizzare il container amministratore solo per scenari di sviluppo e test. Non è consigliabile utilizzarlo in ambienti di produzione. Per ulteriori informazioni, consulta Container amministratore su GitHub.

Ulteriori informazioni

Per ulteriori informazioni sull'utilizzo delle AMI Bottlerocket ottimizzate per Amazon EKS, consulta le sezioni seguenti:

- Per ulteriori informazioni su Bottlerocket, consulta la documentazione e le versioni su GitHub.
- Per utilizzare Bottlerocket con gruppi di nodi gestiti, consulta la sezione Gruppi di nodi gestiti.
- Per avviare nodi Bottlerocket autogestiti, consulta la sezione Avvio dei nodi Bottlerocket autogestiti.
- Per recuperare gli ID più recenti delle AMI Bottlerocket ottimizzate per Amazon EKS, consulta la sezione Recupero degli ID AMI Bottlerocket ottimizzate per Amazon EKS.
- Per informazioni dettagliate sul supporto per la conformità, consulta la sezione Supporto alla conformità Bottlerocket.

Recupero degli ID AMI Bottlerocket ottimizzate per Amazon EKS

Puoi recuperare l'ID Amazon Machine Image (AMI) per le AMI ottimizzate per Amazon EKS interrogando l'API AWS Systems Manager Parameter Store. Con questo parametro, non è necessario eseguire una ricerca manuale degli ID AMI ottimizzate per Amazon EKS. Per ulteriori informazioni sull'API Systems Manager Parameter Store, vedere GetParameter. Il principale IAM che usi deve disporre dell'autorizzazione IAM `ssm:GetParameter` per recuperare i metadati dell'AMI ottimizzata per Amazon EKS.

Puoi recuperare l'ID immagine dell'ultima Bottlerocket AMI ottimizzata consigliata per Amazon EKS con il seguente AWS CLI comando utilizzando il `image_id` sottoparametro. Sostituisci *`1.30`* con una versione supportata e *`region-code`* con una regione supportata da Amazon EKS per la quale desideri recuperare l'ID AMI.

```
aws ssm get-parameter --name /aws/service/bottlerocket/aws-k8s-1.30/x86_64/latest/
image_id --region region-code --query "Parameter.Value" --output text
```

Di seguito viene riportato un output di esempio:

```
ami-1234567890abcdef0
```

Supporto alla conformità Bottlerocket

Bottlerocket è conforme alle raccomandazioni definite da varie organizzazioni:

- È stato definito un Benchmark CIS per Bottlerocket. In una configurazione predefinita, l'immagine Bottlerocket presenta la maggior parte dei controlli richiesti dal profilo di configurazione CIS Level 1. Puoi implementare i controlli necessari per un profilo di configurazione CIS di livello 2. Per ulteriori informazioni, consulta Convalida dell'AMI Bottlerocket ottimizzata di Amazon EKS rispetto al benchmark CIS sul blog AWS.
- Il set di funzionalità ottimizzato e la superficie di attacco ridotta fanno sì che le istanze Bottlerocket richiedano meno configurazioni per soddisfare i requisiti PCI DSS. Il Benchmark CIS per Bottlerocket è una risorsa eccellente per le linee guida in materia di rafforzamento e supporta i requisiti per gli standard di configurazione sicuri ai sensi del requisito PCI DSS 2.2. Puoi sfruttare Fluent Bit per supportare i requisiti di registrazione degli audit a livello di sistema operativo ai sensi del requisito PCI DSS 10.2. AWS pubblica periodicamente nuove istanze Bottlerocket (con patch) per aiutarti a soddisfare i requisiti PCI DSS 6.2 (per la versione 3.2.1) e il requisito 6.3.3 (per la versione 4.0).

- Bottlerocket è una funzionalità idonea alla normativa HIPAA autorizzata per l'uso con carichi di lavoro regolamentati sia per Amazon EC2 che per Amazon EKS. Per ulteriori informazioni, consulta il whitepaper Architecting for HIPAA Security and Compliance on Amazon EKS.

AMI Ubuntu Linux ottimizzate per Amazon EKS

Canonical ha collaborato con Amazon EKS per creare AMI del nodo utilizzabili nei cluster.

Canonical fornisce un'immagine del sistema operativo built-for-purpose Kubernetes Node. Questa Ubuntu immagine minimizzata è ottimizzata per Amazon EKS e include il AWS kernel personalizzato sviluppato congiuntamente con. AWS Per ulteriori informazioni, consulta UbuntuAmazon Elastic Kubernetes Service (EKS) e. Avvio dei nodi Ubuntu autogestiti Per informazioni sul supporto, consulta la sezione Software di terze parti delle Domande frequenti sull'assistenza Premium AWS .

AMI Windows ottimizzate per Amazon EKS

Le AMI ottimizzate per Amazon EKS su Windows sono state sviluppate sulla base di Windows Server 2019 e Windows Server 2022. Sono configurate per fungere da immagine di base per i nodi Amazon EKS. Per impostazione predefinita, le AMI includono i componenti seguenti:

- `kubelet`
- `kube-proxy`
- AWS IAM Authenticator per Kubernetes
- `csi-proxy`
- `containerd`

> **Note**
>
> Puoi monitorare gli eventi di sicurezza o privacy per Windows Server con la Guida all'aggiornamento della sicurezza Microsoft.

Amazon ECS offre AMI ottimizzate per container Windows nelle seguenti varianti:

- AMI Windows Server 2019 Core ottimizzata per Amazon EKS
- AMI Windows Server 2019 Full ottimizzata per Amazon EKS
- AMI Windows Server 2022 Core ottimizzata per Amazon EKS

- AMI Windows Server 2022 Full ottimizzata per Amazon EKS

> **Important**
>
> - L'AMI Windows Server 20H2 Core ottimizzata per Amazon EKS è stata dichiarata obsoleta. Non verranno rilasciate nuove versioni di questa AMI.
> - Per garantire di disporre degli ultimi aggiornamenti di sicurezza per impostazione predefinita, Amazon EKS mantiene Windows AMI ottimizzate negli ultimi 4 mesi. Ogni nuova AMI sarà disponibile per 4 mesi dal momento del rilascio iniziale. Dopo questo periodo, le AMI precedenti vengono rese private e non sono più accessibili. Consigliamo di utilizzare le AMI più recenti per evitare vulnerabilità di sicurezza e perdere l'accesso alle AMI più vecchie che hanno raggiunto la fine del periodo di validità del supporto. Sebbene non possiamo garantire di poter fornire l'accesso alle AMI che sono state rese private, puoi richiedere l'accesso compilando un ticket a. AWS Support

Calendario dei rilasci

Nella tabella seguente sono elencate le date di rilascio e di fine supporto delle versioni Windows su Amazon EKS. Se una data di fine supporto è vuota, la versione è ancora supportata.

Versione Windows	Versione Amazon EKS	Fine del supporto Amazon EKS
Windows Server 2022 Core	10/17/2022	
Windows Server 2022 Full	10/17/2022	
Windows Server 20H2 Core	8/12/2021	8/9/2022
Windows Server 2004 Core	8/19/2020	12/14/2021
Windows Server 2019 Core	10/7/2019	
Windows Server 2019 Full	10/7/2019	
Windows Server 1909 Core	10/7/2019	12/8/2020

Parametri di configurazione dello script di bootstrap

Quando crei un nodo Windows, sul nodo è presente uno script che consente la configurazione di diversi parametri. A seconda della configurazione, questo script può essere presente sul nodo in una posizione simile a: `C:\Program Files\Amazon\EKS\Start-EKSBootstrap.ps1`. È possibile specificare valori di parametri personalizzati specificandoli come argomenti dello script di bootstrap. Ad esempio, puoi aggiornare i dati utente nel modello di avvio. Per ulteriori informazioni, consulta Dati utente Amazon EC2.

Lo script include i seguenti parametri della riga di comando:

- `-EKSClusterName`: specifica il nome del cluster Amazon EKS a cui il nodo worker deve essere aggiunto.
- `-KubeletExtraArgs`: specifica argomenti aggiuntivi per `kubelet` (facoltativo).
- `-KubeProxyExtraArgs`: specifica argomenti aggiuntivi per `kube-proxy` (facoltativo).
- `-APIServerEndpoint`: specifica l'endpoint del server API del cluster Amazon EKS (facoltativo). Valido solo se usato con `-Base64ClusterCA`. Bypassa la chiamata a `Get-EKSCluster`.
- `-Base64ClusterCA`: specifica il contenuto CA del cluster codificato base64 (facoltativo). Valido solo se usato con `-APIServerEndpoint`. Bypassa la chiamata a `Get-EKSCluster`.
- `-DNSClusterIP`: sostituisce l'indirizzo IP da utilizzare per le query DNS all'interno del cluster (facoltativo). L'impostazione predefinita è `10.100.0.10` o `172.20.0.10` in base all'indirizzo IP dell'interfaccia primaria.
- `-ServiceCIDR`: sostituisce l'intervallo di indirizzi IP del servizio Kubernetes da cui vengono indirizzati i servizi del cluster. L'impostazione predefinita è `172.20.0.0/16` o `10.100.0.0/16` in base all'indirizzo IP dell'interfaccia primaria.
- `-ExcludedSnatCIDRs`: un elenco di CIDR `IPv4` da escludere dal Source Network Address Translation (SNAT). Ciò significa che l'IP privato del pod indirizzabile in VPC non verrebbe tradotto nell'indirizzo IP dell'indirizzo `IPv4` primario dell'interfaccia di rete elastica (ENI) dell'istanza per il traffico in uscita. Per impostazione predefinita, viene aggiunto il CIDR `IPv4` del VPC per il nodo Windows Amazon EKS. Se si specificano i CIDR per questo parametro, si escludono anche i CIDR specificati. Per ulteriori informazioni, consulta SNAT per Pods.

Oltre ai parametri della riga di comando, è anche possibile specificare alcuni parametri delle variabili di ambiente. Quando si specifica un parametro della riga di comando, questo ha la precedenza sulla rispettiva variabile di ambiente. Le variabili di ambiente devono essere definite come ambito di computer (o sistema) perché lo script bootstrap leggerà solo le variabili con ambito di computer.

Lo script tiene conto delle seguenti variabili di ambiente:

- `SERVICE_IPV4_CIDR`: per la definizione, fare riferimento al parametro della riga di comando `ServiceCIDR`.
- `EXCLUDED_SNAT_CIDRS`: dovrebbe essere una stringa separata da virgole. Fare riferimento al parametro della riga di comando `ExcludedSnatCIDRs` per la definizione.

Avvio di nodi Windows Server 2022 autogestiti con **`eksctl`**

Puoi utilizzare il seguente **`test-windows-2022.`**`yaml` come riferimento per eseguire i nodi Windows Server 2022 come nodi autogestiti. Sostituisci ogni *`example value`* con i valori in tuo possesso.

> **Note**
>
> Per eseguire nodi Windows Server 2022 autogestiti, è necessario utilizzare `eksctl` versione `0.116.0` o una versione successiva.

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: windows-2022-cluster
  region: region-code
  version: '1.30'

nodeGroups:
  - name: windows-ng
    instanceType: m5.2xlarge
    amiFamily: WindowsServer2022FullContainer
    volumeSize: 100
    minSize: 2
    maxSize: 3
  - name: linux-ng
    amiFamily: AmazonLinux2
    minSize: 2
    maxSize: 3
```

È quindi possibile creare i gruppi di nodi utilizzando il comando seguente.

```
eksctl create cluster -f test-windows-2022.yaml
```

Supporto dell'autenticazione gMSA

I Pods Windows di Amazon EKS consentono diversi tipi di autenticazione dell'account del servizio gestito (gMSA).

- Amazon EKS supporta le identità di dominio Active Directory per l'autenticazione. Per ulteriori informazioni su gMSA aggiunto al dominio, consulta l'articolo Autenticazione di Windows sui podsWindows di Amazon EKS del blog AWS .
- Amazon EKS offre un plug-in che consente ai non-domain-joined Windows nodi di recuperare gMSA le credenziali con un'identità utente portatile. Per ulteriori informazioni su gMSA senza dominio, consulta l'articolo Autenticazione di Windows senza dominio per i podsWindows Amazon EKS del blog AWS .

Immagini di container memorizzate nella cache

Le AMI ottimizzate per Windows di Amazon EKS dispongono di alcune immagini di container memorizzate nella cache per il `containerd` runtime. Le immagini dei container vengono memorizzate nella cache quando si creano AMI personalizzate utilizzando componenti di build gestiti da Amazon. Per ulteriori informazioni, consulta Utilizzo del componente di compilazione gestito da Amazon.

Le seguenti immagini di container memorizzate nella cache sono per il runtime `containerd`:

- `amazonaws.com/eks/pause-windows`
- `mcr.microsoft.com/windows/nanoserver`
- `mcr.microsoft.com/windows/servercore`

Ulteriori informazioni

Per ulteriori informazioni sull'utilizzo delle AMI Windows ottimizzate per Amazon EKS, consulta le sezioni seguenti:

- Per utilizzare Windows con gruppi di nodi gestiti, consulta la sezione Gruppi di nodi gestiti.
- Per avviare nodi Windows autogestiti, consulta la sezione Avvio dei nodi Windows autogestiti.

- Per informazioni sulla versione, consulta Versioni delle AMI Windows ottimizzate per Amazon EKS.
- Per recuperare gli ID più recenti delle AMI Windows ottimizzate per Amazon EKS, consulta la sezione Recupero degli ID AMI Windows ottimizzate per Amazon EKS.
- Per utilizzare Amazon EC2 Image Builder per creare AMI Windows ottimizzate e personalizzate per Amazon EKS, consulta la sezione Creazione di AMI Windows personalizzate ottimizzate per Amazon EKS.
- Per le best practice, consulta la gestione delle Windows AMI ottimizzata per Amazon EKS nella EKS Best Practices Guide.

Versioni delle AMI Windows ottimizzate per Amazon EKS

Important

L'Extended Support per le Windows AMI ottimizzate per Amazon EKS pubblicate da non AWS è disponibile per la Kubernetes versione`1.23`, ma è disponibile per le Kubernetes versioni `1.24` e successive.

Questo argomento elenca le versioni delle Windows AMI ottimizzate per Amazon EKS e le versioni corrispondenti di `kubelet`, `containerd`, e `csi-proxy`.

I metadati di un'AMI ottimizzata per Amazon EKS, incluso l'ID AMI per ogni variante, possono essere recuperati a livello di programmazione. Per ulteriori informazioni, consulta Recupero degli ID AMI Windows ottimizzate per Amazon EKS.

Le versioni dell'AMI si basano sulla versione di Kubernetes e la data di rilascio dell'AMI è nel formato seguente:

```
k8s_major_version.k8s_minor_version-release_date
```

Note

I gruppi di nodi gestiti di Amazon EKS supportano le versioni di novembre 2022 e successive delle Windows AMI.

AMI Windows Server 2022 Core ottimizzata per Amazon EKS

Le tabelle seguenti riportano le versioni correnti e precedenti dell'AMI Windows Server 2022 Core ottimizzata per Amazon EKS.

Kubernetes version 1.30

Kubernetes versione **1.30**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1,29

Kubernetes versione **1.29**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	`containerd` è stato aggiornato a `1.7.11`. `kubelet` è stato aggiornato a `1.29.3`.
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	`containerd` è stato aggiornato a `1.6.28`. CNI ricostruito e in uso. `csi-proxy` `golang 1.22.1`
1.29-2024.03.12	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Risolto un bug per cui l'immagine di pausa veniva erroneamente eliminata dal kubelet processo di raccolta dei rifiuti.
1.29-2024.01.11	1.29.0	1.6.18	1.1.2	Escluso l'Windowsaggiornamento standalone KB5034439 sulle AMI Core di Server 2022. Windows La KB si applica solo alle Windows installazioni con una WinRE partizione separata, che non sono incluse in nessuna delle nostre Windows AMI ottimizzate per Amazon EKS.

Kubernetes version 1,28

Kubernetes versione **1.28**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.28-2024.03.12	1.28.5	1.6.18	1.1.2	
1.28-2024.02.13	1.28.5	1.6.18	1.1.2	
1.28-2024.01.11	1.28.5	1.6.18	1.1.2	WindowsAggiornamento standalone KB5034439 escluso sulle AMI Core di Server 2022. Windows La KB si applica solo alle Windows installazioni con una WinRE partizione separata, che non sono incluse in nessuna delle nostre Windows AMI ottimizzate per Amazon EKS.
1.28-2023.12.12	1.28.3	1.6.18	1.1.2	
1.28-2023.11.14	1.28.3	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.28-2023.10.19	1.28.2	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).
1.28-2023-09.27	1.28.2	1.6.6	1.1.2	Risolto un avviso di sicurezza in kubelet.
1.28-2023.09.12	1.28.1	1.6.6	1.1.2	

Kubernetes version 1.27

Kubernetes versione **1.27**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.27.12.
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.27-2024.03.12	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.11	1.27.9	1.6.18	1.1.2	WindowsAggiornamento standalone KB5034439 escluso sulle AMI Core di Server 2022. Windows La KB si applica solo alle Windows installazioni con una WinRE partizione separata, che non sono incluse in nessuna delle nostre Windows AMI ottimizzate per Amazon EKS.
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Include patch per CVE-2023-5528 .

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Risolto un avviso di sicurezza in kubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	È stato risolto il problema #2042 di containers-roadmap , che impediva ai nodi di estrarre immagini private di Amazon ECR.
1.27-2023.05.17	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetes versione **1.26**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.26.15.
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.26-2024.03.12	1.26.12	1.6.18	1.1.2	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.11	1.26.12	1.6.18	1.1.2	WindowsAggiornamento standalone KB5034439 escluso sulle AMI Core di Server 2022. Windows La KB si applica solo alle Windows installazioni con una WinRE partizione separata, che non sono incluse in nessuna delle nostre Windows AMI ottimizzate per Amazon EKS.
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.26.9. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.26.4. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

Kubernetes versione **1.25**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28.
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
				in uso. `csi-proxy` `golang 1.22.1`
`1.25-2024.03.12`	`1.25.16`	`1.6.18`	`1.1.2`	
`1.25-2024.02.13`	`1.25.16`	`1.6.18`	`1.1.2`	
`1.25-2024.01.11`	`1.25.16`	`1.6.18`	`1.1.2`	WindowsAggiornamento standalone KB5034439 escluso sulle AMI Core di Server 2022. Windows La KB si applica solo alle Windows installazioni con una WinRE partizione separata, che non sono incluse in nessuna delle nostre Windows AMI ottimizzate per Amazon EKS.
`1.25-2023.12.12`	`1.25.15`	`1.6.18`	`1.1.2`	
`1.25-2023.11.14`	`1.25.15`	`1.6.18`	`1.1.2`	Include patch per `CVE-2023-5528` .
`1.25-2023.10.19`	`1.25.14`	`1.6.18`	`1.1.2`	`containerd` è stato aggiornato a `1.6.18`. `kubelet` è stato aggiornato a `1.25.14`. Aggiunte nuove variabili di ambiente dello script di bootstrap (`SERVICE_IPV4_CIDR` e `EXCLUDED_SNAT_CIDRS`).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.25.9. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	È stato installato un pluginMSA senza dominio per facilitare l'autenticazione gMSA per i container Windows su Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetes versione **1.24**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28.
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.24-2024.03.12	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.11	1.24.17	1.6.18	1.1.2	WindowsAggiornamento standalone KB5034439 escluso sulle AMI Core di Server 2022. Windows La KB si applica solo alle Windows installazioni con una WinRE partizione separata, che non sono incluse in nessuna delle nostre Windows AMI ottimizzate per Amazon EKS.
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
				aggiornato a 1.24.17. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.24.13. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	È stato installato un plugin gMSA senza dominio per facilitare l'autenticazione gMSA per i container Windows su Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Versione di Kubernetes declassata a 1.24.7 perché 1.24.10 presenta un problema segnalato in kube-proxy .
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2022.12.13	1.24.7	1.6.6	1.1.1	
1.24-2022.10.11	1.24.7	1.6.6	1.1.1	

AMI Windows Server 2022 Full ottimizzata per Amazon EKS

Le tabelle seguenti riportano le versioni correnti e precedenti dell'AMI Windows Server 2022 Full ottimizzata per Amazon EKS.

Kubernetes version 1,30

Kubernetes versione **1.30**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1,29

Kubernetes versione **1.29**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	`containerd` è stato aggiornato a `1.7.11`. `kubelet` è stato aggiornato a `1.29.3`.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.29-2024.03.12	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Risolto un bug per cui l'immagine di pausa veniva erroneamente eliminata dal kubelet processo di raccolta dei rifiuti.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1.28

Kubernetes versione **1.28**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
				in uso. `csi-proxy` `golang 1.22.1`
`1.28-2024.03.12`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2024.02.13`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2024.01.09`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2023.12.12`	`1.28.3`	`1.6.18`	`1.1.2`	
`1.28-2023.11.14`	`1.28.3`	`1.6.18`	`1.1.2`	Include patch per `CVE-2023-5528` .
`1.28-2023.10.19`	`1.28.2`	`1.6.18`	`1.1.2`	`containerd` è stato aggiornato a `1.6.18`. Aggiunte nuove variabili di ambiente dello script di bootstrap (`SERVICE_IPV4_CIDR` e `EXCLUDED_SNAT_CIDRS`).
`1.28-2023-09.27`	`1.28.2`	`1.6.6`	`1.1.2`	Risolto un avviso di sicurezza in `kubelet`.
`1.28-2023.09.12`	`1.28.1`	`1.6.6`	`1.1.2`	

Kubernetes version 1.27

Kubernetes versione **1.27**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.27.12.
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.27-2024.03.12	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Risolto un avviso di sicurezza in kubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	È stato risolto il problema #2042 di containers-roadmap , che impediva ai nodi di estrarre immagini private di Amazon ECR.
1.27-2023.05.18	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetes versione **1.26**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.26.15.
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.26-2024.03.12	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.26.9. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.26.4. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

Kubernetes versione **1.25**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28.
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.25-2024.03.12	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.25.14. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.25.9. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	È stato installato un pluginMSA senza dominio per facilitare l'autenticazione gMSA per i container Windows su Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetes versione **1.24**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28.
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.24-2024.03.12	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.24.17. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.24.13. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	È stato installato un pluginMSA senza dominio per facilitare l'autenticazione gMSA per i container Windows su Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Versione di Kubernetes declassata a 1.24.7 perché 1.24.10 presenta un problema segnalato in kube-proxy .
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2022.12.14	1.24.7	1.6.6	1.1.1	
1.24-2022.10.11	1.24.7	1.6.6	1.1.1	

AMI Windows Server 2019 Core ottimizzata per Amazon EKS

Le tabelle seguenti riportano le versioni correnti e precedenti dell'AMI Windows Server 2019 Core ottimizzata per Amazon EKS.

Kubernetes version 1.30

Kubernetes versione **1.30**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1,29

Kubernetes versione **1.29**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	`containerd` è stato aggiornato a `1.7.11`. `kubelet` è stato aggiornato a `1.29.3`.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Risolto un bug per cui l'immagine di pausa veniva erroneamente eliminata dal kubelet processo di raccolta dei rifiuti.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1.28

Kubernetes versione **1.28**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
				in uso. `csi-proxy` `golang 1.22.1`
`1.28-2024.03.13`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2024.02.13`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2024.01.09`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2023.12.12`	`1.28.3`	`1.6.18`	`1.1.2`	
`1.28-2023.11.14`	`1.28.3`	`1.6.18`	`1.1.2`	Include patch per `CVE-2023-5528` .
`1.28-2023.10.19`	`1.28.2`	`1.6.18`	`1.1.2`	`containerd` è stato aggiornato a `1.6.18`. Aggiunte nuove variabili di ambiente dello script di bootstrap (`SERVICE_IPV4_CIDR` e `EXCLUDED_SNAT_CIDRS`).
`1.28-2023-09.27`	`1.28.2`	`1.6.6`	`1.1.2`	Risolto un avviso di sicurezza in `kubelet`.
`1.28-2023.09.12`	`1.28.1`	`1.6.6`	`1.1.2`	

Kubernetes version 1.27

Kubernetes versione **1.27**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.27.12.
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.27-2024.03.13	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Risolto un avviso di sicurezza in kubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	È stato risolto il problema #2042 di containers-roadmap , che impediva ai nodi di estrarre immagini private di Amazon ECR.
11.27-2023.05.18	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetes versione **1.26**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.26.15.
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.26-2024.03.13	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.26.9. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.26.4. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

Kubernetes versione **1.25**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28.
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.25-2024.03.13	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.25.14. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.25.9. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	È stato installato un pluginMSA senza dominio per facilitare l'autenticazione gMSA per i container Windows su Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetes versione **1.24**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28.
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.24-2024.03.13	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.24.17. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.20	1.24.13	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.24.13. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	È stato installato un pluginMSA senza dominio per facilitare l'autenticazione gMSA per i container Windows su Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Versione di Kubernetes declassata a 1.24.7 perché 1.24.10 presenta un problema segnalato in kube-proxy .
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2022.12.13	1.24.7	1.6.6	1.1.1	
1.24-2022.11.08	1.24.7	1.6.6	1.1.1	

AMI Windows Server 2019 Full ottimizzata per Amazon EKS

Le tabelle seguenti riportano le versioni correnti e precedenti dell'AMI Windows Server 2019 Full ottimizzata per Amazon EKS.

Kubernetes version 1.30

Kubernetes versione **1.30**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.30-2024.05.15	1.30.0	1.6.28	1.1.2	

Kubernetes version 1,29

Kubernetes versione **1.29**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.29-2024.05.15	1.29.3	1.7.11	1.1.2	`containerd` è stato aggiornato a `1.7.11`. `kubelet` è stato aggiornato a `1.29.3`.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.29-2024.04.09	1.29.0	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.29-2024.03.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.13	1.29.0	1.6.25	1.1.2	
1.29-2024.02.06	1.29.0	1.6.25	1.1.2	Risolto un bug per cui l'immagine di pausa veniva erroneamente eliminata dal kubelet processo di raccolta dei rifiuti.
1.29-2024.01.09	1.29.0	1.6.18	1.1.2	

Kubernetes version 1.28

Kubernetes versione **1.28**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.28-2024.05.14	1.28.8	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.28.8.
1.28-2024.04.09	1.28.5	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
				in uso. `csi-proxy` `golang 1.22.1`
`1.28-2024.03.13`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2024.02.13`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2024.01.09`	`1.28.5`	`1.6.18`	`1.1.2`	
`1.28-2023.12.12`	`1.28.3`	`1.6.18`	`1.1.2`	
`1.28-2023.11.14`	`1.28.3`	`1.6.18`	`1.1.2`	Include patch per `CVE-2023-5528` .
`1.28-2023.10.19`	`1.28.2`	`1.6.18`	`1.1.2`	`containerd` è stato aggiornato a `1.6.18`. Aggiunte nuove variabili di ambiente dello script di bootstrap (`SERVICE_IPV4_CIDR` e `EXCLUDED_SNAT_CIDRS`).
`1.28-2023-09.27`	`1.28.2`	`1.6.6`	`1.1.2`	Risolto un avviso di sicurezza in `kubelet`.
`1.28-2023.09.12`	`1.28.1`	`1.6.6`	`1.1.2`	

Kubernetes version 1.27

Kubernetes versione **1.27**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2024.05.14	1.27.12	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.27.12.
1.27-2024.04.09	1.27.9	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.27-2024.03.13	1.27.9	1.6.18	1.1.2	
1.27-2024.02.13	1.27.9	1.6.18	1.1.2	
1.27-2024.01.09	1.27.9	1.6.18	1.1.2	
1.27-2023.12.12	1.27.7	1.6.18	1.1.2	
1.27-2023.11.14	1.27.7	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.27-2023.10.19	1.27.6	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2023-09.27	1.27.6	1.6.6	1.1.2	Risolto un avviso di sicurezza in kubelet.
1.27-2023.09.12	1.27.4	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.27-2023.08.17	1.27.4	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.27-2023.08.08	1.27.3	1.6.6	1.1.1	
1.27-2023.07.11	1.27.3	1.6.6	1.1.1	
1.27-2023.06.20	1.27.1	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.27-2023.06.14	1.27.1	1.6.6	1.1.1	È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.27-2023.06.06	1.27.1	1.6.6	1.1.1	È stato risolto il problema #2042 di containers-roadmap , che impediva ai nodi di estrarre immagini private di Amazon ECR.
1.27-2023.05.17	1.27.1	1.6.6	1.1.1	

Kubernetes version 1.26

Kubernetes versione **1.26**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2024.05.14	1.26.15	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28. kubelet è stato aggiornato a 1.26.15.
1.26-2024.04.09	1.26.12	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.26-2024.03.13	1.26.12	1.6.18	1.1.2	
1.26-2024.02.13	1.26.12	1.6.18	1.1.2	
1.26-2024.01.09	1.26.12	1.6.18	1.1.2	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2023.12.12	1.26.10	1.6.18	1.1.2	
1.26-2023.11.14	1.26.10	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.26-2023.10.19	1.26.9	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.26.9. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).
1.26-2023.09.12	1.26.7	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.26-2023.08.17	1.26.7	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.26-2023.08.08	1.26.6	1.6.6	1.1.1	
1.26-2023.07.11	1.26.6	1.6.6	1.1.1	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.26-2023.06.20	1.26.4	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.26-2023.06.14	1.26.4	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.26.4. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.
1.26-2023.05.09	1.26.2	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.26-2023.04.26	1.26.2	1.6.6	1.1.1	
1.26-2023.04.11	1.26.2	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.26-2023.03.24	1.26.2	1.6.6	1.1.1	

Kubernetes version 1,25

Kubernetes versione **1.25**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2024.05.14	1.25.16	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28.
1.25-2024.04.09	1.25.16	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.25-2024.03.13	1.25.16	1.6.18	1.1.2	
1.25-2024.02.13	1.25.16	1.6.18	1.1.2	
1.25-2024.01.09	1.25.16	1.6.18	1.1.2	
1.25-2023.12.12	1.25.15	1.6.18	1.1.2	
1.25-2023.11.14	1.25.15	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.25-2023.10.19	1.25.14	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.25.14. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2023.09.12	1.25.12	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.25-2023.08.17	1.25.12	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.25-2023.08.08	1.25.9	1.6.6	1.1.1	
1.25-2023.07.11	1.25.9	1.6.6	1.1.1	
1.25-2023.06.20	1.25.9	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.25-2023.06.14	1.25.9	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.25.9. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.25-2023.05.09	1.25.7	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.25-2023.04.11	1.25.7	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.25-2023.03.27	1.25.6	1.6.6	1.1.1	È stato installato un pluginMSA senza dominio per facilitare l'autenticazione gMSA per i container Windows su Amazon EKS.
1.25-2023.03.20	1.25.6	1.6.6	1.1.1	
1.25-2023.02.14	1.25.6	1.6.6	1.1.1	

Kubernetes version 1.24

Kubernetes versione **1.24**

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2024.05.14	1.24.17	1.6.28	1.1.2	containerd è stato aggiornato a 1.6.28.
1.24-2024.04.09	1.24.17	1.6.25	1.1.2	containerd è stato aggiornato a 1.6.25. CNI ricostruito e in uso. csi-proxy golang 1.22.1
1.24-2024.03.13	1.24.17	1.6.18	1.1.2	
1.24-2024.02.13	1.24.17	1.6.18	1.1.2	
1.24-2024.01.09	1.24.17	1.6.18	1.1.2	
1.24-2023.12.12	1.24.17	1.6.18	1.1.2	
1.24-2023.11.14	1.24.17	1.6.18	1.1.2	Include patch per CVE-2023-5528 .
1.24-2023.10.19	1.24.17	1.6.18	1.1.2	containerd è stato aggiornato a 1.6.18. kubelet è stato aggiornato a 1.24.17. Aggiunte nuove variabili di ambiente dello script di bootstrap (SERVICE_IPV4_CIDR e EXCLUDED_SNAT_CIDRS).

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2023.09.12	1.24.16	1.6.6	1.1.2	È stato aggiornato il plugin Amazon VPC CNI per utilizzare il file binario del connettore Kubernetes, che ottiene Pod l'indirizzo IP dal server API Kubernetes. La richiesta di pull #100 è stata unita.
1.24-2023.08.17	1.24.16	1.6.6	1.1.2	Include patch per CVE-2023-3676 , CVE-2023-3893 e CVE-2023-3955 .
1.24-2023.08.08	1.24.13	1.6.6	1.1.1	
1.24-2023.07.11	1.24.13	1.6.6	1.1.1	
1.24-2023.06.21	1.24.13	1.6.6	1.1.1	È stato risolto il problema che causava la compilazione errata dell'elenco di ricerca dei suffissi DNS.
1.24-2023.06.14	1.24.13	1.6.6	1.1.1	Kubernetes è stato aggiornato a 1.24.13. È stato aggiunto il supporto per la mappatura delle porte host in CNI. La richiesta di pull #93 è stata unita.

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2023.05.09	1.24.7	1.6.6	1.1.1	È stato risolto un bug che causava il problema di connettività di rete #1126 sui pod dopo il riavvio del nodo. È stato introdotto un nuovo parametro di configurazione dello script di bootstrap (ExcludedSnatCIDRs).
1.24-2023.04.11	1.24.7	1.6.6	1.1.1	È stato aggiunto un meccanismo di ripristino per kubelet e kube-proxy in caso di arresto anomalo del servizio.
1.24-2023.03.27	1.24.7	1.6.6	1.1.1	È stato installato un pluginMSA senza dominio per facilitare l'autenticazione gMSA per i container Windows su Amazon EKS.
1.24-2023.03.20	1.24.7	1.6.6	1.1.1	Versione di Kubernetes declassata a 1.24.7 perché 1.24.10 presenta un problema segnalato in kube-proxy .
1.24-2023.02.14	1.24.10	1.6.6	1.1.1	
1.24-2023.01.23	1.24.7	1.6.6	1.1.1	
1.24-2023.01.11	1.24.7	1.6.6	1.1.1	

Versione AMI	Versione **kubelet**	Versione **containerd**	Versione **csi-proxy**	Note di rilascio
1.24-2022.12.14	1.24.7	1.6.6	1.1.1	
1.24-2022.10.12	1.24.7	1.6.6	1.1.1	

Recupero degli ID AMI Windows ottimizzate per Amazon EKS

Puoi recuperare a livello di codice l'ID Amazon Machine Image (AMI) per le AMI ottimizzate per Amazon EKS interrogando l'API Parameter Store. AWS Systems Manager Questo parametro consente di evitare la ricerca manuale degli ID AMI ottimizzata per Amazon EKS. Per ulteriori informazioni sull'API Systems Manager Parameter Store, vedere GetParameter. Il principale IAM che usi deve disporre dell'autorizzazione IAM `ssm:GetParameter` per recuperare i metadati dell'AMI ottimizzata per Amazon EKS.

È possibile recuperare l'ID immagine della versione consigliata dell'AMI Windows ottimizzata per Amazon EKS più recente con il comando seguente utilizzando il parametro secondario `image_id`. È possibile sostituire *`1.30`* con qualsiasi versione supportata da Amazon EKS ed è possibile sostituire *`region-code`* con una regione supportata da Amazon EKS per la quale si desidera ottenere l'ID AMI. Sostituisci *`Core`* con `Full` per visualizzare l'ID AMI di Windows Server Full. Per Kubernetes versione `1.24` o successiva, puoi sostituire *`2019`* con `2022` per visualizzare l'ID AMI Windows Server 2022.

```
aws ssm get-parameter --name /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-EKS_Optimized-1.30/image_id --region region-code --query "Parameter.Value" --output text
```

Di seguito viene riportato un output di esempio:

```
ami-1234567890abcdef0
```

Creazione di AMI Windows personalizzate ottimizzate per Amazon EKS

Puoi utilizzare EC2 Image Builder per creare AMI Windows ottimizzate e personalizzate per Amazon EKS con una delle seguenti opzioni:

- Utilizzo di un'AMI Windows ottimizzata per Amazon EKS come base
- Utilizzo del componente di compilazione gestito da Amazon

In entrambi i casi, è necessario creare la propria ricetta di Image Builder. Per ulteriori informazioni, consulta la sezione Creazione di una nuova versione di una ricetta di immagine nella Guida per l'utente di Image Builder.

> **Important**
>
> I seguenti componenti gestiti da Amazon per `eks` includono patch per `CVE-2023-5528`.
>
> - `1.24.3` e versioni successive
> - `1.25.2` e versioni successive
> - `1.26.2` e versioni successive
> - `1.27.0` e versioni successive
> - `1.28.0` e versioni successive

Utilizzo di un'AMI Windows ottimizzata per Amazon EKS come base

Questa opzione è il modo consigliato per creare AMI Windows personalizzate. Le AMI Windows ottimizzate per Amazon EKS che forniamo vengono aggiornate più frequentemente rispetto al componente di compilazione gestito da Amazon.

1. Inizia a creare la tua ricetta di Image Builder.
   a. Apri la console EC2 Image Builder all'indirizzo https://console.aws.amazon.com/imagebuilder.
   b. Nel riquadro di navigazione di sinistra, scegli Ricette di immagini.
   c. Scegli Crea ricetta di immagine.
2. Nella sezione Dettagli ricetta, inserisci un nome e una versione.
3. Specifica l'ID dell'AMI Windows ottimizzata per Amazon EKS nella sezione Immagine di base.

a. Scegli Inserisci un ID dell'AMI personalizzata.

b. Recupera l'ID dell'AMI per la versione del sistema operativo Windows richiesta. Per ulteriori informazioni, consulta Recupero degli ID AMI Windows ottimizzate per Amazon EKS.

c. Inserisci l'ID dell'AMI personalizzata. Se l'ID AMI non viene trovato, assicurati che l'ID Regione AWS per l'AMI corrisponda a quello Regione AWS mostrato nella parte superiore destra della console.

4. (Facoltativo) Per ottenere gli ultimi aggiornamenti di sicurezza, aggiungi il componente `update-windows` nella sezione Componenti di compilazione.

a. Dall'elenco a discesa a destra della casella di ricerca Trova componenti per nome scegli Gestito da Amazon.

b. Nella sezione Trova componenti per nome, inserisci **`update-windows`**.

c. Seleziona la casella di controllo del risultato della ricerca **`update-windows`**. Il componente include le patch di Windows più recenti per il sistema operativo.

5. Completa gli input rimanenti della ricetta di immagine con le configurazioni richieste. Per ulteriori informazioni, consulta la sezione Creazione di una nuova versione di una ricetta di immagine (console) nella Guida per l'utente di Image Builder.

6. Scegli Crea ricetta.

7. Utilizza la nuova ricetta di immagine in una pipeline di immagini nuova o esistente. Una volta che la pipeline di immagini viene eseguita correttamente, l'AMI personalizzata verrà elencata come immagine di output e sarà pronta per l'uso. Per ulteriori informazioni, consulta la sezione Creazione di una pipeline di immagini utilizzando la console di EC2 Image Builder.

Utilizzo del componente di compilazione gestito da Amazon

Quando non è possibile utilizzare un'AMI Windows ottimizzata per Amazon EKS come base, puoi utilizzare il componente di compilazione gestito da Amazon. Questa opzione potrebbe essere in ritardo rispetto alle versioni di Kubernetes supportate più recenti.

1. Inizia a creare la tua ricetta di Image Builder.

a. Apri la console EC2 Image Builder all'indirizzo https://console.aws.amazon.com/imagebuilder.

b. Nel riquadro di navigazione di sinistra, scegli Ricette di immagini.

c. Scegli Crea ricetta di immagine.

2. Nella sezione Dettagli della ricetta, inserisci un nome e una versione.
3. Determina quale opzione utilizzerai per creare la tua AMI personalizzata nella sezione Immagine di base:
   - Seleziona immagini gestite: scegli Windows come Sistema operativo (OS) per le immagini. Quindi scegli una delle seguenti opzioni per Origine dell'immagine.
     - Quick start (gestito da Amazon): nell'elenco a discesa Nome immagine, seleziona una versione di Windows Server supportata da Amazon EKS. Per ulteriori informazioni, consulta AMI Windows ottimizzate per Amazon EKS.
     - Immagini di mia proprietà: per Nome immagine, seleziona l'ARN della tua immagine con la tua licenza. L'immagine fornita non deve avere componenti Amazon EKS già installati.
   - Inserisci l'ID dell'AMI personalizzata: per l'ID dell'AMI, inserisci l'ID della tua AMI con la tua licenza. L'immagine fornita non deve avere componenti Amazon EKS già installati.
4. Nella sezione Componenti di compilazione - Windows, esegui le seguenti operazioni:
   a. Dall'elenco a discesa a destra della casella di ricerca Trova componenti per nome scegli Gestito da Amazon.
   b. Nella casella di ricerca Trova componenti per nome, inserisci **`eks`**.
   c. Seleziona la casella di controllo accanto al risultato della ricerca **`eks-optimized-ami-windows`**, anche se il risultato restituito potrebbe non essere la versione desiderata.
   d. Nella casella di ricerca Trova componenti per nome, inserisci **`update-windows`**.
   e. Seleziona la casella di controllo accanto al risultato della ricerca update-windows. Il componente include le patch di Windows più recenti per il sistema operativo.
5. Nella sezione Componenti selezionati, esegui le seguenti operazioni:
   a. Scegli Opzioni di controllo delle versioni per **`eks-optimized-ami-windows`**.
   b. Scegli Specifica la versione del componente.
   c. Nel campo Versione del componente, inserisci ***`version.x`*** , sostituendo *`version`* con una versione di Kubernetes supportata. L'immissione di un *`x`* parziale del numero di versione indica di utilizzare la versione più recente del componente che si allinea anche alla parte della versione definita in modo esplicito. Presta attenzione all'output della console, in quanto ti avviserà se la versione desiderata è disponibile come componente gestito. Tieni presente che le versioni di Kubernetes più recenti potrebbero non essere disponibili per il componente di compilazione. Per ulteriori informazioni sulle versioni disponibili, consulta Recupero di informazioni sulle versioni dei componenti di `eks-optimized-ami-windows`.

> **Note**
>
> Le seguenti versioni dei componenti di compilazione di `eks-optimized-ami-windows` richiedono una versione di `eksctl` `0.129` o inferiore:
>
> - `1.24.0`

6. Completa gli input rimanenti della ricetta di immagine con le configurazioni richieste. Per ulteriori informazioni, consulta la sezione Creazione di una nuova versione di una ricetta di immagine (console) nella Guida per l'utente di Image Builder.
7. Scegli Crea ricetta.
8. Utilizza la nuova ricetta di immagine in una pipeline di immagini nuova o esistente. Una volta che la pipeline di immagini viene eseguita correttamente, l'AMI personalizzata verrà elencata come immagine di output e sarà pronta per l'uso. Per ulteriori informazioni, consulta la sezione Creazione di una pipeline di immagini utilizzando la console di EC2 Image Builder.

Recupero di informazioni sulle versioni dei componenti di **`eks-optimized-ami-windows`**

È possibile recuperare informazioni specifiche relative a ciò che viene installato con ciascun componente. Ad esempio, puoi verificare quale versione di `kubelet` è installata. I componenti sono sottoposti a test funzionali sulle versioni di sistema operativo Windows supportate da Amazon EKS. Per ulteriori informazioni, consulta Calendario dei rilasci. Altre eventuali versioni del sistema operativo Windows non elencate come supportate o che hanno raggiunto la fine del supporto potrebbero non essere compatibili con il componente.

1. Apri la console EC2 Image Builder all'indirizzo https://console.aws.amazon.com/imagebuilder.
2. Nel riquadro di navigazione a sinistra, scegli Componenti.
3. Dall'elenco a discesa a destra della casella di ricerca Trova componenti per nome, modifica la selezione Di mia proprietà con Quick start (gestito da Amazon).
4. Nella sezione Trova componenti per nome, inserire **`eks`**.
5. (Facoltativo) Se stai utilizzando una versione recente, ordina la colonna Versione in ordine decrescente selezionandola due volte.
6. Scegli il collegamento **`eks-optimized-ami-windows`** con la versione desiderata.

La Descrizione nella pagina risultante mostra le informazioni specifiche.

Archiviazione

Questo capitolo illustra le opzioni di archiviazione per i cluster Amazon EKS.

Argomenti

Driver CSI per Amazon EBS

Il driver Container Storage Interface (CSI) per Amazon Elastic Block Store (Amazon EBS) gestisce il ciclo di vita dei volumi Amazon EBS come storage per i volumi Kubernetes creati. Il driver Amazon EBS CSI crea volumi Amazon EBS per questi tipi di volumi: Kubernetes volumi effimeri generici e volumi persistenti.

Ecco alcuni aspetti da considerare quando si utilizza il driver CSI di Amazon EBS.

- Il plug-in Amazon EBS CSI richiede le autorizzazioni IAM per effettuare chiamate alle AWS API per tuo conto. Per ulteriori informazioni, consulta Creazione del ruolo IAM per il driver CSI per Amazon EBS.
- Non puoi montare volumi Amazon EBS su Fargate Pods.
- Puoi eseguire il controller Amazon EBS CSI sui nodi Fargate, ma sul nodo Amazon EBS CSI DaemonSet può essere eseguito solo su istanze Amazon EC2.

Il driver CSI per Amazon EBS non viene installato quando si crea un cluster per la prima volta. Per utilizzare il driver, è necessario aggiungerlo come componente aggiuntivo Amazon EKS o come componente aggiuntivo autogestito.

- Per istruzioni su come aggiungerlo come componente aggiuntivo Amazon EKS, consultare Gestione del driver CSI per Amazon EBS come componente aggiuntivo Amazon EKS.
- Per istruzioni su come aggiungerlo come installazione autogestita, consulta il progetto Driver CSI (Container Storage Interface) di Amazon EBS su GitHub.

Dopo aver installato il driver CSI secondo uno dei due metodi, è possibile testare la funzionalità con un'applicazione di esempio. Per ulteriori informazioni, consulta Implementare un'applicazione di esempio e verificare il funzionamento del driver CSI.

Creazione del ruolo IAM per il driver CSI per Amazon EBS

Il plug-in Amazon EBS CSI richiede le autorizzazioni IAM per effettuare chiamate alle AWS API per tuo conto. Per ulteriori informazioni, consulta Configurazione dell'autorizzazione driver su GitHub.

> **Note**
>
> I Pods avranno accesso alle autorizzazioni assegnate al ruolo IAM, a meno che non si blocchi l'accesso a IMDS. Per ulteriori informazioni, consulta Best practice di sicurezza per Amazon EKS.

Prerequisiti

- Un cluster esistente.
- Un provider AWS Identity and Access Management (IAM) OpenID Connect (OIDC) esistente per il tuo cluster. Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.

La procedura seguente mostra come creare un ruolo IAM e collegarci la policy gestita da AWS . Puoi utilizzare `eksctl`, la AWS Management Console o la AWS CLI.

> **Note**
>
> I passaggi specifici di questa procedura sono stati scritti per l'utilizzo del driver come componente aggiuntivo di Amazon EKS. Per utilizzare il driver come componente aggiuntivo autogestito occorre una procedura differente. Per ulteriori informazioni, consulta Configurazione delle autorizzazioni del driver su GitHub.

eksctl

Creazione del ruolo IAM del plugin CSI per Amazon EBS con **`eksctl`**

1. Crea un ruolo IAM e allega una policy. AWS mantiene una politica AWS gestita oppure puoi creare la tua politica personalizzata. È possibile creare un ruolo IAM e allegare la policy AWS gestita con il seguente comando. Sostituisci *`my-cluster`* con il nome del cluster. Il comando distribuisce uno AWS CloudFormation stack che crea un ruolo IAM e vi allega la policy IAM. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws: arn:aws-us-gov:`

   ```
   eksctl create iamserviceaccount \
       --name ebs-csi-controller-sa \
       --namespace kube-system \
       --cluster my-cluster \
       --role-name AmazonEKS_EBS_CSI_DriverRole \
       --role-only \
       --attach-policy-arn arn:aws:iam::aws:policy/service-role/
   AmazonEBSCSIDriverPolicy \
       --approve
   ```

2. Se utilizzi una chiave KMS personalizzata per la crittografia dei volumi Amazon EBS, personalizza il ruolo IAM secondo necessità. Ad esempio, completa le seguenti operazioni:

   a. Copia e incolla il codice seguente in un nuovo file *`kms-key-for-encryption-on-ebs`*`.json`. Sostituisci *`custom-key-arn`* con l'ARN della chiave KMS personalizzata.

      ```
      {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Action": [
              "kms:CreateGrant",
              "kms:ListGrants",
              "kms:RevokeGrant"
            ],
            "Resource": ["custom-key-arn"],
            "Condition": {
              "Bool": {
                "kms:GrantIsForAWSResource": "true"
              }
      ```

```
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

b. Crea la policy. È possibile modificare *KMS_Key_For_Encryption_On_EBS_Policy* con un nome diverso. Se esegui questa operazione, tuttavia, assicurati di modificare il nome anche nelle fasi successive.

```
aws iam create-policy \
  --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
  --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Collega la policy IAM al ruolo con il comando seguente. Sostituisci *111122223333* con l'ID del tuo account. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/KMS_Key_For_Encryption_On_EBS_Policy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

AWS Management Console

Per creare il ruolo IAM del plug-in Amazon EBS CSI con AWS Management Console

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.

3. Nella pagina Ruoli, seleziona Crea ruolo.
4. Nella pagina Select trusted entity (Seleziona entità attendibile), esegui le operazioni seguenti:
   a. Nella sezione Tipo di identità attendibile, scegli Identità Web.
   b. Per Identity provider (Provider di identità), scegli l'URL del provider OpenID Connect per il cluster (come riportato in Overview [Panoramica] in Amazon EKS).
   c. Per Pubblico, scegli `sts.amazonaws.com`.
   d. Seleziona Avanti.
5. Nella pagina Add permissions (Aggiungi autorizzazioni), esegui le operazioni seguenti:
   a. Nella casella Filtra policy, inserisci `AmazonEBSCSIDriverPolicy`.
   b. Seleziona la casella di controllo a sinistra della `AmazonEBSCSIDriverPolicy` restituita dalla ricerca.
   c. Seleziona Avanti.
6. Nella pagina Name, review, and create (Assegna un nome, rivedi e crea), esegui le operazioni seguenti:
   a. Per Role name (Nome ruolo), inserisci un nome univoco per il ruolo, ad esempio ***`AmazonEKS_EBS_CSI_DriverRole`***.
   b. In Aggiungi tag (facoltativo), aggiungi metadati al ruolo collegando i tag come coppie chiave-valore. Per ulteriori informazioni sull'utilizzo di tag in IAM, consulta la sezione Applicazione di tag alle risorse IAM nella Guida per l'utente di IAM.
   c. Scegli Crea ruolo.
7. Dopo aver creato il ruolo, sceglilo nella console in modo da aprirlo per la modifica.
8. Scegli la scheda Relazioni di attendibilità e quindi Modifica policy di attendibilità.
9. Trova la riga simile alla seguente:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

   Inserisci una virgola alla fine della riga precedente, quindi aggiungine una seguente dopo quella precedente. *`region-code`*Sostituiscilo con quello in Regione AWS cui si trova il tuo cluster. Sostituisci *`EXAMPLED539D4633E53DE1B71EXAMPLE`* con l'ID provider OIDC del cluster.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:ebs-csi-controller-sa"
```

10. Scegli Aggiorna policy per concludere.
11. Se utilizzi una chiave KMS personalizzata per la crittografia dei volumi Amazon EBS, personalizza il ruolo IAM secondo necessità. Ad esempio, completa le seguenti operazioni:
    a. Nel pannello di navigazione a sinistra, seleziona Policy.
    b. Nella pagina Policy, scegli Crea policy.
    c. Nella pagina Crea policy, scegli la scheda JSON.
    d. Copia e incolla il codice seguente nell'editor, sostituendo *custom-key-arn* con l'ARN della chiave KMS personalizzata.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
```

```
    }
  ]
}
```

e. Scegliere Successivo: Tag.

f. Nella pagina Add tags (optional) (Aggiungi tag, facoltativo), seleziona Next: Review (Successivo: esamina).

g. Per Nome immettere un nome univoco per la policy (ad esempio ***KMS_Key_For_Encryption_On_EBS_Policy***).

h. Scegli Crea policy.

i. Nel pannello di navigazione a sinistra, seleziona Ruoli.

j. Scegli ***DriverRoleAmazoneKS_EBS_CSI_*** nella console per aprirlo e modificarlo.

k. Dall'elenco a discesa Aggiungi autorizzazioni, scegli Collega policy.

l. Nella casella Filtra policy, inserisci *KMS_Key_For_Encryption_On_EBS_Policy*.

m. Seleziona la casella di controllo a sinistra della *KMS_Key_For_Encryption_On_EBS_Policy* restituita dalla ricerca.

n. Scegli Collega policy.

AWS CLI

Per creare il ruolo IAM del plug-in Amazon EBS CSI con AWS CLI

1. Visualizza l'URL del provider OIDC del cluster. Sostituire *my-cluster* con il nome del cluster. Se l'output dal comando è `None`, rivedi i Prerequisiti.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

Di seguito viene riportato un output di esempio:

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Crea il ruolo IAM concedendo l'autorizzazione per l'operazione `AssumeRoleWithWebIdentity`.

a. Copia i contenuti seguenti in un file denominato *aws-ebs-csi-driver-trust-policy*.json. Sostituisci *111122223333* con l'ID del tuo account. Sostituisci

EXAMPLED539D4633E53DE1B71EXAMPLE e *region-code* con i valori restituiti nella fase precedente. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringEquals": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:ebs-csi-controller-sa"
        }
      }
    }
  ]
}
```

b. Crea il ruolo. È possibile modificare *AmazonEKS_EBS_CSI_DriverRole* con un nome diverso. Se modifichi il valore, assicurati di modificarlo anche nelle fasi successive.

```
aws iam create-role \
  --role-name AmazonEKS_EBS_CSI_DriverRole \
  --assume-role-policy-document file://"aws-ebs-csi-driver-trust-
policy.json"
```

3. Allega una policy. AWS mantiene una politica AWS gestita oppure è possibile creare una politica personalizzata. AWS Associa la politica gestita al ruolo con il seguente comando. Se il cluster si trova negli AWS GovCloud Stati Uniti orientali o AWS GovCloud negli Stati Uniti occidentali Regioni AWS, `arn:aws:` sostituiscilo con. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
```

```
--policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy \
--role-name AmazonEKS_EBS_CSI_DriverRole
```

4. Se utilizzi una chiave KMS personalizzata per la crittografia dei volumi Amazon EBS, personalizza il ruolo IAM secondo necessità. Ad esempio, completa le seguenti operazioni:

   a. Copia e incolla il codice seguente in un nuovo file *kms-key-for-encryption-on-ebs*.json. Sostituisci *custom-key-arn* con l'ARN della chiave KMS personalizzata.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
      ],
      "Resource": ["custom-key-arn"],
      "Condition": {
        "Bool": {
          "kms:GrantIsForAWSResource": "true"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
      ],
      "Resource": ["custom-key-arn"]
    }
  ]
}
```

   b. Crea la policy. È possibile modificare *KMS_Key_For_Encryption_On_EBS_Policy* con un nome diverso. Se esegui questa operazione, tuttavia, assicurati di modificare il nome anche nelle fasi successive.

```
aws iam create-policy \
  --policy-name KMS_Key_For_Encryption_On_EBS_Policy \
  --policy-document file://kms-key-for-encryption-on-ebs.json
```

c. Collega la policy IAM al ruolo con il comando seguente. Sostituisci *111122223333* con l'ID del tuo account. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/KMS_Key_For_Encryption_On_EBS_Policy \
  --role-name AmazonEKS_EBS_CSI_DriverRole
```

Ora che hai creato il ruolo IAM del driver CSI di Amazon EBS, puoi passare a Aggiunta del componente aggiuntivo CSI per Amazon EBS. Quando implementi il plugin tramite questa procedura, crea ed è configurato per l'utilizzo di un account di servizio denominato `ebs-csi-controller-sa`. L'account di servizio è associato a un `clusterrole` Kubernetes a cui vengono assegnate le autorizzazioni Kubernetes necessarie.

Gestione del driver CSI per Amazon EBS come componente aggiuntivo Amazon EKS

Per migliorare la sicurezza e ridurre il numero di attività, è possibile gestire il driver CSI per Amazon EBS come componente aggiuntivo. Per ulteriori informazioni sui componenti aggiuntivi Amazon EKS, consultare Componenti aggiuntivi Amazon EKS. È possibile aggiungere il componente aggiuntivo CSI per Amazon EBS seguendo la procedura descritta in Aggiunta del componente aggiuntivo CSI per Amazon EBS.

Se è stato aggiunto il componente aggiuntivo CSI per Amazon EBS, sarà possibile gestirlo seguendo la procedura descritta nelle sezioni Aggiornamento del driver CSI per Amazon EBS come componente aggiuntivo Amazon EKS e Rimozione del componente aggiuntivo CSI per Amazon EBS.

Prerequisiti

- Un cluster esistente. Per visualizzare la versione della piattaforma richiesta, esegui il comando seguente.

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver
```

- Un provider OpenID Connect (OIDC) AWS Identity and Access Management (IAM) esistente per il cluster. Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- Ruolo IAM del driver CSI per Amazon EBS. Se non soddisfi questo prerequisito, il tentativo di installare il componente aggiuntivo e l'esecuzione di `kubectl describe pvc` riporteranno `failed to provision volume with StorageClass` insieme a un errore `could not create volume in EC2: UnauthorizedOperation`. Per ulteriori informazioni, consulta Creazione del ruolo IAM per il driver CSI per Amazon EBS.
- Se utilizzi un cluster con `PodSecurityPolicy` con restrizioni a livello di cluster, assicurati che al componente aggiuntivo siano concesse autorizzazioni sufficienti per essere implementato. Per le autorizzazioni richieste da ciascun componente aggiuntivoPod, vedere la definizione del relativo manifesto del componente aggiuntivo su. GitHub

Important

Per utilizzare la funzionalità snapshot del driver CSI per Amazon EBS, prima del componente aggiuntivo è necessario installare il dispositivo esterno di acquisizione snapshot. I componenti dei dispositivi esterni di acquisizione snapshot devono essere installati secondo l'ordine seguente:

- `CustomResourceDefinition` (CRD) per `volumesnapshotclasses`, `volumesnapshots` e `volumesnapshotcontents`
- RBAC (`ClusterRole`, `ClusterRoleBinding` e così via)
- implementazione del controller

Per ulteriori informazioni, consulta Dispositivo esterno di acquisizione snapshot CSI su GitHub.

Aggiunta del componente aggiuntivo CSI per Amazon EBS

> **Important**
>
> Prima di aggiungere il driver Amazon EBS come componente aggiuntivo di Amazon EKS, verifica che nel cluster non sia installata una versione autogestita del driver. In tal caso, vediDisinstallazione di un driver Amazon EBS CSI autogestitosulGitHub.

Per aggiungere il componente aggiuntivo CSI per Amazon EBS al cluster, puoi utilizzare `eksctl`, la AWS Management Console o la AWS CLI.

eksctl

Per aggiungere il componente aggiuntivo CSI per Amazon EBS utilizzando **`eksctl`**

Esegui il comando seguente. Sostituisci *`my-cluster`* con il nome del cluster, *`111122223333`* con il tuo ID account e *`AmazonEKS_EBS_CSI_DriverRole`* con il nome del ruolo IAM creato in precedenza. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali)Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

```
eksctl create addon --name aws-ebs-csi-driver --cluster my-cluster --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --force
```

Se si rimuove l'opzione **`--force`** e tutte le impostazioni del componente aggiuntivo Amazon EKS sono in conflitto con le impostazioni esistenti, l'aggiornamento del componente aggiuntivo Amazon EKS non avrà successo. Verrà quindi visualizzato un messaggio di errore con la procedura utile a risolvere il conflitto. Prima di specificare questa opzione, assicurati che il componente aggiuntivo di Amazon EKS non gestisca le impostazioni che devi gestire tu, perché questa opzione le sovrascrive. Per ulteriori informazioni su altre opzioni per questa impostazione, consulta Componenti aggiuntivi nella documentazione di `eksctl`. Per ulteriori informazioni sulla gestione dei campi Kubernetes di Amazon EKS, consulta Gestione dei campi Kubernetes.

AWS Management Console

Per aggiungere il componente aggiuntivo CSI per Amazon EBS utilizzando la AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.

2. Nel pannello di navigazione a sinistra, seleziona Cluster.
3. Scegli il nome del cluster per cui configurare il componente aggiuntivo CSI per Amazon EBS.
4. Seleziona la scheda Componenti aggiuntivi.
5. Scegli Ottieni altri componenti aggiuntivi.
6. Nella pagina Seleziona componenti aggiuntivi, procedi come segue:
    a. Nella sezione Componenti aggiuntivi di Amazon EKS, seleziona la casella di controllo Driver CSI di Amazon EBS.
    b. Seleziona Avanti.
7. Nella pagina Configura le impostazioni dei componenti aggiuntivi selezionati, procedi come segue:
    a. Seleziona la Versione che desideri utilizzare.
    b. In Seleziona ruolo IAM, seleziona il nome del ruolo IAM a cui è stata collegata la policy IAM del driver CSI di Amazon EBS.
    c. (Facoltativo) È possibile espandere le Impostazioni di configurazione facoltative. Se selezioni Sostituisci per Metodo di risoluzione dei conflitti, una o più impostazioni per il componente aggiuntivo esistente possono essere sovrascritte con le impostazioni del componente aggiuntivo di Amazon EKS. Se rimuovi questa opzione e c'è un conflitto con le impostazioni esistenti, l'operazione non va a buon fine e viene visualizzato un messaggio di errore per aiutarti a risolvere il conflitto. Prima di selezionare questa opzione, assicurarsi che il componente aggiuntivo Amazon EKS non gestisca le impostazioni che devono essere gestite dall'utente.
    d. Seleziona Avanti.
8. Nella pagina Rivedi e aggiungi, scegli Crea. Una volta completata l'installazione, viene visualizzato il componente aggiuntivo.

AWS CLI

Per aggiungere il componente aggiuntivo CSI per Amazon EBS utilizzando la AWS CLI

Esegui il comando seguente. Sostituire *my-cluster* con il nome del cluster, *111122223333* con il proprio ID account e *AmazonEKS_EBS_CSI_DriverRole* con il nome del ruolo creato in precedenza. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali)Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole
```

Ora che hai aggiunto il driver CSI per Amazon EBS come componente aggiuntivo di Amazon EKS, puoi passare a Implementare un'applicazione di esempio e verificare il funzionamento del driver CSI. Questa procedura comprende la configurazione della classe di archiviazione.

Aggiornamento del driver CSI per Amazon EBS come componente aggiuntivo Amazon EKS

Amazon EKS non aggiorna automaticamente il CSI per Amazon EBS nel cluster quando vengono rilasciate nuove versioni o dopo l'aggiornamento del cluster a una nuova versione secondaria di Kubernetes. Per aggiornare il CSI per Amazon EBS su un cluster esistente, avviare l'aggiornamento, quindi Amazon EKS aggiornerà automaticamente il componente aggiuntivo.

eksctl

Per aggiornare il componente aggiuntivo CSI per Amazon EBS utilizzando **`eksctl`**

1. Verificare la versione corrente del componente aggiuntivo CSI per Amazon EBS. Sostituire *`my-cluster`* con il nome del cluster.

   ```
   eksctl get addon --name aws-ebs-csi-driver --cluster my-cluster
   ```

   Di seguito viene riportato un output di esempio:

   ```
   NAME                    VERSION                     STATUS  ISSUES  IAMROLE
    UPDATE AVAILABLE
   aws-ebs-csi-driver      v1.11.2-eksbuild.1          ACTIVE  0
     v1.11.4-eksbuild.1
   ```

2. Aggiornare il componente aggiuntivo alla versione restituita in `UPDATE AVAILABLE` nell'output della fase precedente.

   ```
   eksctl update addon --name aws-ebs-csi-driver --version v1.11.4-eksbuild.1 --
   cluster my-cluster \
     --service-account-role-arn
    arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --force
   ```

Se si rimuove l'opzione `--force` e tutte le impostazioni del componente aggiuntivo Amazon EKS sono in conflitto con le impostazioni esistenti, l'aggiornamento del componente aggiuntivo Amazon EKS non avrà successo. Verrà quindi visualizzato un messaggio di errore con la procedura utile a risolvere il conflitto. Prima di specificare questa opzione, assicurati che il componente aggiuntivo di Amazon EKS non gestisca le impostazioni che devi gestire tu, perché questa opzione le sovrascrive. Per ulteriori informazioni su altre opzioni per questa impostazione, consulta Componenti aggiuntivi nella documentazione di `eksctl`. Per ulteriori informazioni sulla gestione dei campi Kubernetes di Amazon EKS, consulta Gestione dei campi Kubernetes.

AWS Management Console

Per aggiornare il componente aggiuntivo CSI per Amazon EBS utilizzando la AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, seleziona Cluster.
3. Scegli il nome del cluster per cui aggiornare il componente aggiuntivo CSI per Amazon EBS.
4. Seleziona la scheda Componenti aggiuntivi.
5. Scegli Driver CSI di Amazon EBS.
6. Scegli Modifica.
7. Nella pagina Configura il driver CSI di Amazon EBS, procedi come segue:
   a. Seleziona la Versione che desideri utilizzare.
   b. In Seleziona ruolo IAM, seleziona il nome del ruolo IAM a cui è stata collegata la policy IAM del driver CSI di Amazon EBS.
   c. (Facoltativo) È possibile espandere le Impostazioni di configurazione facoltative e modificarle secondo necessità.
   d. Seleziona Salvataggio delle modifiche.

AWS CLI

Per aggiornare il componente aggiuntivo CSI per Amazon EBS utilizzando la AWS CLI

1. Verificare la versione corrente del componente aggiuntivo CSI per Amazon EBS. Sostituire *my-cluster* con il nome del cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver
 --query "addon.addonVersion" --output text
```

   Di seguito viene riportato un output di esempio:

```
v1.11.2-eksbuild.1
```

2. Determinazione delle versioni del componente aggiuntivo CSI per Amazon EBS disponibili per la versione del cluster.

```
aws eks describe-addon-versions --addon-name aws-ebs-csi-driver --kubernetes-
version 1.23 \
  --query "addons[].addonVersions[].[addonVersion,
 compatibilities[].defaultVersion]" --output text
```

   Di seguito viene riportato un output di esempio:

```
v1.11.4-eksbuild.1
True
v1.11.2-eksbuild.1
False
```

   La versione con `True` è la versione predefinita implementata al momento della creazione del componente aggiuntivo. La versione implementata al momento della creazione del componente aggiuntivo potrebbe non essere l'ultima versione disponibile. Nell'output precedente, la versione più recente viene implementata quando viene creato il componente aggiuntivo.

3. Aggiornare il componente aggiuntivo alla versione con `True` restituito nell'output della fase precedente. È inoltre possibile eseguire l'aggiornamento a una versione successiva, se restituito nell'output.

```
aws eks update-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver
 --addon-version v1.11.4-eksbuild.1 \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_EBS_CSI_DriverRole --resolve-
conflicts PRESERVE
```

L'opzione *PRESERVE* conserva tutte le impostazioni personalizzate che hai impostato per il componente aggiuntivo. Per ulteriori informazioni su altre opzioni per questa impostazione, consulta update-addon nella documentazione di riferimento della riga di comando Amazon EKS. Per ulteriori informazioni sulla gestione della configurazione dei componenti aggiuntivi di Amazon EKS, consulta Gestione dei campi Kubernetes.

Rimozione del componente aggiuntivo CSI per Amazon EBS

Sono possibili due opzioni per rimuovere un componente aggiuntivo Amazon EKS.

- Mantenere il software aggiuntivo sul cluster – Questa opzione rimuove la gestione di Amazon EKS su qualsiasi impostazione. Inoltre, rimuove la possibilità per Amazon EKS di notificare gli aggiornamenti e aggiornare automaticamente il componente aggiuntivo Amazon EKS dopo l'avvio di un aggiornamento. Tuttavia, mantiene il software aggiuntivo sul cluster. Questa opzione rende il componente aggiuntivo come autogestito anziché come un componente aggiuntivo Amazon EKS. Con questa opzione, non ci sono tempi di inattività per il componente aggiuntivo. I comandi in questa procedura utilizzano questa opzione.
- Rimuovere completamente il software aggiuntivo dal cluster – Consigliamo di rimuovere il componente aggiuntivo Amazon EKS dal cluster solo se non ci sono risorse nel cluster che dipendono dallo stesso. Per eseguire questa operazione, elimina `--preserve` dal comando utilizzato in questa procedura.

Se al componente aggiuntivo è associato un account IAM, l'account IAM non viene rimosso.

Per rimuovere il componente aggiuntivo CSI per Amazon EB, puoi utilizzare `eksctl`, la AWS Management Console o la AWS CLI.

eksctl

Per rimuovere il componente aggiuntivo CSI per Amazon EBS utilizzando **`eksctl`**

Sostituire *my-cluster* con il nome del cluster ed eseguire il comando seguente.

```
eksctl delete addon --cluster my-cluster --name aws-ebs-csi-driver --preserve
```

AWS Management Console

Per rimuovere il componente aggiuntivo CSI per Amazon EBS utilizzando la AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, seleziona Cluster.
3. Scegli il nome del cluster per cui desideri rimuovere il componente aggiuntivo CSI per Amazon EBS.
4. Seleziona la scheda Componenti aggiuntivi.
5. Scegli Driver CSI di Amazon EBS.
6. Scegli Rimuovi.
7. Nella finestra di dialogo di aws-ebs-csi-driver conferma Rimuovi:, procedi come segue:
   a. Se desideri che Amazon EKS interrompa la gestione delle impostazioni per il componente aggiuntivo, seleziona Mantieni sul cluster. Effettuare questa operazione se si desidera conservare il software aggiuntivo sul cluster. In questo modo sarà possibile gestire autonomamente tutte le impostazioni del componente aggiuntivo.
   b. Specificare **`aws-ebs-csi-driver`**.
   c. Selezionare Remove (Rimuovi).

AWS CLI

Per rimuovere il componente aggiuntivo CSI per Amazon EBS utilizzando la AWS CLI

Sostituire *`my-cluster`* con il nome del cluster ed eseguire il comando seguente.

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-ebs-csi-driver --
preserve
```

Implementare un'applicazione di esempio e verificare il funzionamento del driver CSI

È possibile testare la funzionalità del driver CSI con un'applicazione di esempio. In questo argomento viene illustrata una procedura, ma è anche possibile eseguire le operazioni seguenti:

- Implementare un'applicazione di esempio che utilizza il dispositivo esterno di acquisizione snapshot per creare snapshot del volume. Per ulteriori informazioni, consulta Snapshot del volume su GitHub.
- Implementare un'applicazione di esempio che utilizza il ridimensionamento del volume. Per ulteriori informazioni, consulta Ridimensionamento del volume su GitHub.

Questa procedura utilizza l'esempio di provisioning di volumi dinamici disponibile nel repository GitHub del driver Container Storage Interface (CSI) per Amazon EBS finalizzato all'uso di un volume Amazon EBS con provisioning dinamico.

1. Clona il repository GitHub del driver Container Storage Interface (CSI) per Amazon EBS nel sistema locale.

   ```
   git clone https://github.com/kubernetes-sigs/aws-ebs-csi-driver.git
   ```

2. Passare alla directory di esempio `dynamic-provisioning`.

   ```
   cd aws-ebs-csi-driver/examples/kubernetes/dynamic-provisioning/
   ```

3. (Facoltativo) Per impostazione predefinita il file `manifests/storageclass.yaml` effettua il provisioning dei volumi Amazon EBS `gp2`. Per utilizzare invece i volumi `gp3`, aggiungi `type: gp3` a `manifests/storageclass.yaml`.

   ```
   echo "parameters:
     type: gp3" >> manifests/storageclass.yaml
   ```

4. Implementare la classe di storage `ebs-sc`, la dichiarazione di volume persistente `ebs-claim` e l'applicazione di esempio `app` dalla directory `manifests`.

   ```
   kubectl apply -f manifests/
   ```

5. Descrivere la classe di storage `ebs-sc`.

```
kubectl describe storageclass ebs-sc
```

Di seguito viene riportato un output di esempio:

```
Name:            ebs-sc
IsDefaultClass:  No
Annotations:     kubectl.kubernetes.io/last-applied-
configuration={"apiVersion":"storage.k8s.io/v1","kind":"StorageClass","metadata":
{"annotations":{},"name":"ebs-
sc"},"provisioner":"ebs.csi.aws.com","volumeBindingMode":"WaitForFirstConsumer"}

Provisioner:           ebs.csi.aws.com
Parameters:            <none>
AllowVolumeExpansion:  <unset>
MountOptions:          <none>
ReclaimPolicy:         Delete
VolumeBindingMode:     WaitForFirstConsumer
Events:                <none>
```

> **Note**
>
> La classe di storage utilizza la modalità di associazione di volumi `WaitForFirstConsumer`. Ciò significa che i volumi non vengono assegnati dinamicamente finché un Pod non effettua una dichiarazione di volume persistente. Per ulteriori informazioni, consulta la sezione Volume Binding Mode (Modalità di associazione dei volumi) nella documentazione Kubernetes.

6. Controlla i Pods nello spazio dei nomi predefinito. Dopo pochi minuti, lo stato del pod Pod `app` cambia in `Running`.

```
kubectl get pods --watch
```

Inserire `Ctrl+C` per tornare a un prompt della shell.

7. Elencare i volumi persistenti nello spazio dei nomi predefinito. Cercare un volume persistente con la richiesta `default/ebs-claim`.

```
kubectl get pv
```

Di seguito viene riportato un output di esempio:

```
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY
   STATUS   CLAIM               STORAGECLASS   REASON   AGE
pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a   4Gi        RWO            Delete
   Bound    default/ebs-claim   ebs-sc                  30s
```

8. Descrivere il volume persistente. Sostituisci `pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a` con il valore dell'output del passaggio precedente.

```
kubectl describe pv pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a
```

Di seguito viene riportato un output di esempio:

```
Name:              pvc-37717cd6-d0dc-11e9-b17f-06fad4858a5a
Labels:            <none>
Annotations:       pv.kubernetes.io/provisioned-by: ebs.csi.aws.com
Finalizers:        [kubernetes.io/pv-protection external-attacher/ebs-csi-aws-com]
StorageClass:      ebs-sc
Status:            Bound
Claim:             default/ebs-claim
Reclaim Policy:    Delete
Access Modes:      RWO
VolumeMode:        Filesystem
Capacity:          4Gi
Node Affinity:
  Required Terms:
    Term 0:        topology.ebs.csi.aws.com/zone in [region-code]
Message:
Source:
    Type:              CSI (a Container Storage Interface (CSI) volume source)
    Driver:            ebs.csi.aws.com
    VolumeHandle:      vol-0d651e157c6d93445
    ReadOnly:          false
    VolumeAttributes:      storage.kubernetes.io/
csiProvisionerIdentity=1567792483192-8081-ebs.csi.aws.com
Events:                <none>
```

L'ID volume Amazon EBS è il valore per `VolumeHandle` nell'output precedente.

9. Verificare che il Pod scriva correttamente i dati nel volume.

```
kubectl exec -it app -- cat /data/out.txt
```

Di seguito viene riportato un output di esempio:

```
Wed May 5 16:17:03 UTC 2021
Wed May 5 16:17:08 UTC 2021
Wed May 5 16:17:13 UTC 2021
Wed May 5 16:17:18 UTC 2021
[...]
```

10. Al termine, eliminare le risorse per questa applicazione di esempio.

```
kubectl delete -f manifests/
```

Domande frequenti sulla migrazione CSI di Amazon EBS

> **Important**
>
> Se Pods è in esecuzione su un cluster versione `1.22` o precedente, per evitare l'interruzione del servizio dovrai installare il driver CSI di Amazon EBS prima di aggiornare il cluster alla versione `1.23`.

La funzionalità di migrazione del container storage interface (CSI) di Amazon EBS trasferisce la responsabilità della gestione delle operazioni di archiviazione dal provisioner di archiviazione EBS in-tree di Amazon EBS al driver CSI di Amazon EBS.

Cosa sono i driver CSI?

Driver CSI:

- sostituire i driver di archiviazione "predefiniti" di Kubernetes esistenti nel codice sorgente del progetto Kubernetes.
- lavorare con provider di archiviazione, come Amazon EBS.
- Fornisci un modello di plugin semplificato che faciliti ai provider di archiviazione come AWS il rilascio di funzionalità e il mantenimento del supporto senza dipendere dal ciclo di rilascio di Kubernetes.

Per ulteriori informazioni, consulta Introduzione nella documentazione CSI di Kubernetes.

Che cos'è la migrazione CSI?

La funzionalità di migrazione di CSI Kubernetes sposta la responsabilità per la gestione delle operazioni di archiviazione dai plugin di archiviazione predefiniti, ad esempio `kubernetes.io/aws-ebs`, ai corrispondenti driver CSI. Gli oggetti `StorageClass`, `PersistentVolume` e `PersistentVolumeClaim` (PVC) esistenti continueranno a funzionare fintanto che il corrispondente driver CSI è installato. Quando la funzione è abilitata:

- I carichi di lavoro esistenti che utilizzano i PVC continueranno a funzionare come sempre.
- Kubernetes passa il controllo di tutte le operazioni di gestione dell'archiviazione ai driver CSI.

Per ulteriori informazioni, consulta Kubernetes1.23: aggiornamento dello stato di migrazione dai volumi predefiniti Kubernetes a CSI sul blog Kubernetes.

Per migrare dal plugin predefinito ai driver CSI, gli indicatori `CSIMigration` e `CSIMigrationAWS` sono abilitati per impostazione predefinita nei cluster Amazon EKS versione `1.23` e successive. Questi indicatori consentono al cluster di tradurre le API predefinite nelle relative API CSI equivalenti. Questi indicatori sono impostati sul piano di controllo Kubernetes gestito da Amazon EKS e nelle impostazioni `kubelet` configurate nelle AMI ottimizzate per Amazon EKS. Se sono presenti Pods che utilizzano volumi Amazon EBS nel cluster, dovrai installare il driver CSI di Amazon EBS prima di aggiornare il cluster alla versione **`1.23`**. In caso contrario, le operazioni di volume come il provisioning e il montaggio potrebbero non funzionare come previsto. Per ulteriori informazioni, consulta Driver CSI per Amazon EBS.

> **Note**
>
> Il provisioner di `StorageClass` predefinito si chiama `kubernetes.io/aws-ebs`. Il provisioner di CSI `StorageClass` di Amazon EBS si chiama `ebs.csi.aws.com`.

Posso montare i volumi **`kubernetes.io/aws-ebs StorageClass`** sui cluster **`1.23`** e versioni successive?

Sì, purché il driver CSI per Amazon EBS sia installato. Per cluster versione `1.23` e successive appena creati, consigliamo di installare il driver CSI di Amazon EBS come parte del processo

di creazione del cluster. Consigliamo di utilizzare solo `StorageClasses` in base al provisioner `ebs.csi.aws.com`.

Se hai aggiornato il tuo piano di controllo del cluster alla versione `1.23` e non hai ancora aggiornato i nodi a `1.23`, allora gli indicatori `kubelet` `CSIMigration` e `CSIMigrationAWS` non sono abilitati. In questo caso, il driver predefinito viene utilizzato per il montaggio di volumi basati su `kubernetes.io/aws-ebs`. Tuttavia, per garantire che i Pods che utilizzano i volumi basati su `kubernetes.io/aws-ebs` possano essere programmati, è necessario comunque installare il driver CSI di Amazon EBS. Il driver è necessario anche per la riuscita di altre operazioni di volume.

Posso eseguire il provisioning di volumi **`kubernetes.io/aws-ebs StorageClass`** su cluster Amazon EKS **`1.23`** e versioni successive?

Sì, purché il driver CSI per Amazon EBS sia installato.

Il provisioner **`kubernetes.io/aws-ebs StorageClass`** sarà mai rimosso da Amazon EKS?

Il provisioner `kubernetes.io/aws-ebs` `StorageClass` e il tipo di volume `awsElasticBlockStore`non sono più supportati, ma non ci sono piani relativi alla loro rimozione. Queste risorse vengono trattate come parte dell'API Kubernetes.

Come devo installare il driver CSI di Amazon EBS?

Consigliamo di installare il componente aggiuntivo di Amazon EKS per il driver CSI di Amazon EBS. Quando è necessario un aggiornamento per il componente aggiuntivo di Amazon EKS, tu avvii l'aggiornamento e Amazon EKS aggiorna il componente aggiuntivo per tuo conto. Se desideri gestire autonomamente il driver, puoi installarlo utilizzando il grafico Helm open source.

> **Important**
>
> Il driver Amazon EBS predefinito Kubernetes viene eseguito sul piano di controllo Kubernetes. Utilizza le autorizzazioni IAM assegnate al Ruolo IAM del cluster Amazon EKS per eseguire il provisioning di volumi Amazon EBS. Il driver CSI di Amazon EBS viene eseguito sui nodi. Il driver necessita delle autorizzazioni IAM per eseguire il provisioning dei volumi. Per ulteriori informazioni, consulta Creazione del ruolo IAM per il driver CSI per Amazon EBS.

Come posso verificare se il driver CSI di Amazon EBS è installato nel mio cluster?

Per stabilire se il driver è installato nel cluster, esegui questo comando:

```
kubectl get csidriver ebs.csi.aws.com
```

Per verificare se l'installazione è gestita da Amazon EKS, esegui questo comando:

```
aws eks list-addons --cluster-name my-cluster
```

Amazon EKS impedirà un aggiornamento del cluster alla versione **1.23** se non ho già installato il driver CSI di Amazon EBS?

No.

Cosa succede se dimentico di installare il driver CSI di Amazon EBS prima di aggiornare il cluster alla versione 1.23? Posso installare il driver dopo aver aggiornato il mio cluster?

Sì, ma le operazioni di volume che richiedono il driver CSI di Amazon EBS dopo l'aggiornamento del cluster avranno esito negativo finché il driver non sarà installato.

Qual è la **StorageClass** predefinita applicata nei cluster Amazon EKS versione **1.23** e successive appena creati?

Il comportamento della `StorageClass` predefinita rimarrà invariato. Con ogni nuovo cluster, Amazon EKS applica una `StorageClass` basata su `kubernetes.io/aws-ebs` denominata `gp2`. Non abbiamo intenzione di rimuovere questa `StorageClass` dai cluster appena creati. Separata dalla `StorageClass` del cluster predefinito, se crei una `StorageClass` basata su `ebs.csi.aws.com` senza specificare un tipo di volume, il driver CSI di Amazon EBS utilizzerà automaticamente `gp3`.

Amazon EKS apporterà modifiche alle **StorageClasses** già presenti nel mio cluster quando aggiorno il mio cluster alla versione **1.23**?

No.

Come posso eseguire la migrazione di un volume permanente dalla **`StorageClasskubernetes.io/aws-ebs`** a **`ebs.csi.aws.com`** tramite gli snapshot?

Per eseguire la migrazione di un volume permanente, consulta Migrazione di cluster Amazon EKS da volumi EBS gp2 a gp3 sul blog AWS.

Come posso modificare un volume Amazon EBS utilizzando le annotazioni?

A partire da `aws-ebs-csi-driver` `v1.19.0-eksbuild.2`, puoi modificare i volumi Amazon EBS utilizzando le annotazioni all'interno dei rispettivi `PersistentVolumeClaim` (PVC). La nuova funzionalità di modifica del volume è implementata come sidecar aggiuntivo, chiamato `volumemodifier`. Per ulteriori informazioni, consulta Semplificazione della migrazione e modifica dei volumi Amazon EBS su Kubernetes utilizzando il driver CSI per EBS sul blog AWS.

La migrazione è supportata per i carichi di lavoro Windows?

Sì. Se stai installando il driver CSI di Amazon EBS utilizzando il grafico Helm open source, imposta `node.enableWindows` su `true`. Questa è l'impostazione predefinita se installi il driver CSI per Amazon EBS come componente aggiuntivo di Amazon EKS. Durante la creazione di `StorageClasses`, imposta il `fsType` su un file system di Windows, ad esempio `ntfs`. Le operazioni di volume per i carichi di lavoro Windows vengono quindi migrate al driver CSI di Amazon EBS nello stesso modo in cui lo sono per i carichi di lavoro Linux.

Driver CSI per Amazon EFS

Amazon Elastic File System (Amazon EFS) fornisce un'archiviazione di file serverless e completamente elastica in modo da poter condividere i dati dei file senza dover fornire o gestire la capacità e le prestazioni di archiviazione. Il driver Amazon EFS Container Storage Interface (CSI) fornisce un'interfaccia CSI che consente Kubernetes ai cluster in esecuzione di AWS gestire il ciclo di vita dei file system Amazon EFS. Questo argomento illustra come implementare il driver CSI per Amazon EBS nel cluster Amazon EKS.

Considerazioni

- Il driver CSI per Amazon EFS non è compatibile con le immagini container basate su Windows.
- Non è possibile utilizzare il provisioning dinamico per volumi persistenti con i nodi Fargate, ma è possibile utilizzare il provisioning statico.

- Il provisioning dinamico richiede `1.2` o una versione successiva del driver. Puoi utilizzare il provisioning statico per volumi persistenti utilizzando la versione `1.1` del driver su qualsiasi versione del cluster Amazon EKS supportata.
- La versione `1.3.2` o successiva di questo driver supporta l'architettura Arm64, incluse le istanze basate su Graviton di Amazon EC2.
- La versione `1.4.2` o successiva di questo driver supporta l'uso di FIPS per il montaggio di file system.
- Osserva le quote delle risorse per Amazon EFS. Ad esempio, è possibile creare una quota di 1.000 punti di accesso per ogni file system Amazon EFS. Per ulteriori informazioni, consulta le Quote di risorse di Amazon EFS che non puoi modificare.

Prerequisiti

- Un provider AWS Identity and Access Management (IAM) OpenID Connect (OIDC) esistente per il tuo cluster. Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:

> **Note**
>
> Un file system Amazon EFS Pod in esecuzione monta AWS Fargate automaticamente un file system Amazon EFS.

Creazione di un ruolo IAM

Il driver CSI per Amazon EFS richiede le autorizzazioni IAM per interagire con il file system. Crea un ruolo IAM e associa ad esso la policy AWS gestita richiesta. Puoi utilizzare `eksctl`, la AWS Management Console o la AWS CLI.

> **Note**
>
> I passaggi specifici di questa procedura sono stati scritti per l'utilizzo del driver come componente aggiuntivo di Amazon EKS. Per informazioni dettagliate sulle installazioni autogestite, consulta Set up driver permission su GitHub.

eksctl

Creazione del ruolo IAM del plugin CSI per Amazon EFS con **`eksctl`**

Per creare il ruolo IAM, eseguire i comandi seguenti. Sostituisci *`my-cluster`* con il nome del cluster e *`AmazonEKS_EFS_CSI_DriverRole`* con il nome del ruolo.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEFSCSIDriverPolicy \
    --approve
TRUST_POLICY=$(aws iam get-role --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
```

```
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

AWS Management Console

Per creare il ruolo IAM del driver CSI di Amazon EFS con AWS Management Console

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Nella pagina Ruoli, seleziona Crea ruolo.
4. Nella pagina Select trusted entity (Seleziona entità attendibile), esegui le operazioni seguenti:
   a. Nella sezione Tipo di identità attendibile, scegli Identità Web.
   b. Per Identity provider (Provider di identità), scegli l'URL del provider OpenID Connect per il cluster (come riportato in Overview [Panoramica] in Amazon EKS).
   c. Per Pubblico, scegli `sts.amazonaws.com`.
   d. Seleziona Successivo.
5. Nella pagina Add permissions (Aggiungi autorizzazioni), esegui le operazioni seguenti:
   a. Nella casella Filtra policy, inserisci *`AmazonEFSCSIDriverPolicy`*.
   b. Seleziona la casella di controllo a sinistra della *`AmazonEFSCSIDriverPolicy`* restituita dalla ricerca.
   c. Seleziona Successivo.
6. Nella pagina Name, review, and create (Assegna un nome, rivedi e crea), esegui le operazioni seguenti:
   a. Per Role name (Nome ruolo), inserisci un nome univoco per il ruolo, ad esempio ***`AmazonEKS_EFS_CSI_DriverRole`***.
   b. In Aggiungi tag (facoltativo), aggiungi metadati al ruolo collegando i tag come coppie chiave-valore. Per ulteriori informazioni sull'utilizzo di tag in IAM, consulta la sezione Applicazione di tag alle risorse IAM nella Guida per l'utente di IAM.
   c. Scegli Crea ruolo.
7. Dopo aver creato il ruolo, sceglilo nella console in modo da aprirlo per la modifica.
8. Scegli la scheda Relazioni di attendibilità e quindi Modifica policy di attendibilità.
9. Trova la riga simile alla seguente:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Aggiungi la riga seguente sopra la riga precedente. *region-code*Sostituiscilo con Regione AWS quello in cui si trova il cluster. Sostituisci *EXAMPLED539D4633E53DE1B71EXAMPLE* con il gestore di ID OIDC del tuo cluster.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:efs-csi-*",
```

10. Modifica l'operatore `Condition` da `"StringEquals"` a `"StringLike"`.
11. Scegli Aggiorna policy per concludere.

AWS CLI

Per creare il ruolo IAM del driver CSI di Amazon EFS con AWS CLI

1. Visualizza l'URL del provider OIDC del cluster. Sostituire *my-cluster* con il nome del cluster. Se l'output dal comando è `None`, rivedi i Prerequisiti.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

Di seguito viene riportato un output di esempio:

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Crea il ruolo IAM che concede l'autorizzazione per l'operazione `AssumeRoleWithWebIdentity`.

   a. Copia i contenuti seguenti in un file denominato *aws-efs-csi-driver-trust-policy*`.json`. Sostituisci *111122223333* con l'ID del tuo account. Sostituisci *EXAMPLED539D4633E53DE1B71EXAMPLE* e *region-code* con i valori restituiti nella fase precedente. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringLike": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:efs-csi-*",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. Crea il ruolo. È possibile modificare *AmazonEKS_EFS_CSI_DriverRole* con un nome diverso. In tal caso, assicurati di modificarlo anche nelle fasi successive.

```
aws iam create-role \
  --role-name AmazonEKS_EFS_CSI_DriverRole \
  --assume-role-policy-document file://"aws-efs-csi-driver-trust-
policy.json"
```

3. Allega la politica AWS gestita richiesta al ruolo con il comando seguente. Se il cluster si trova negli AWS GovCloud Stati Uniti orientali o AWS GovCloud negli Stati Uniti occidentali Regioni AWS, `arn:aws:` sostituiscilo con. `arn:aws-us-gov:`

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy \
  --role-name AmazonEKS_EFS_CSI_DriverRole
```

Installazione del driver Amazon EFS CSI

Consigliamo di installare il driver Amazon EFS CSI attraverso il componente aggiuntivo di Amazon EKS. Per aggiungere un componente aggiuntivo di Amazon EKS al cluster, consulta Creazione di un componente aggiuntivo. Per ulteriori informazioni sui componenti aggiuntivi, consulta Componenti aggiuntivi Amazon EKS. Se non riesci a utilizzare il componente aggiuntivo di Amazon EKS, ti invitiamo a segnalare il problema in Roadmap dei container GitHub.

In alternativa, se desideri un'installazione autogestita del driver Amazon EFS CSI, consulta Installazionesu GitHub.

Creazione di un file system Amazon EFS

Per creare un file system Amazon EFS, consulta Creazione di un sistema di file Amazon EFS per Amazon EKS su GitHub.

Implementazione di un'applicazione di esempio

Puoi implementare diverse app di esempio e modificarle in base alle tue esigenze. Per ulteriori informazioni, consulta Esempi su GitHub.

Driver CSI per Amazon FSx for Lustre

Il driver Container Storage Interface (CSI) per FSx for Lustre mette a disposizione un'interfaccia CSI che permette ai cluster Amazon EKS di gestire il ciclo di vita dei file system FSx for Lustre. Per ulteriori informazioni, consulta la Guida per l'utente di FSx per Lustre.

Questo argomento illustra come implementare il driver CSI per FSx for Lustre nel cluster Amazon EKS e come verificarne il funzionamento. Ti consigliamo di utilizzare sempre la versione più recente del driver. Per le versioni disponibili, vedere la matrice di compatibilità delle specifiche CSI su GitHub.

> **Note**
>
> Il driver non è supportato su Fargate.

Per una descrizione dettagliata dei parametri disponibili e per esempi completi che illustrano le caratteristiche del driver, consulta il progetto Driver Container Storage Interface (CSI) per FSx per Lustre su GitHub.

Prerequisiti

È necessario avere:

- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.
- La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:

Le procedure seguenti consentono di creare un semplice cluster di test con il driver CSI di FSx per Lustre per osservarne il funzionamento. Ti consigliamo di non utilizzare il cluster di test per i carichi di lavoro di produzione. Per questo tutorial ti consigliamo di utilizzare i *`example values`*, tranne quando è indicato di sostituirli. È possibile sostituire qualsiasi *`example value`* quando si completano i passaggi per il cluster di produzione. Ti consigliamo di completare tutti i passaggi nello stesso terminale, dato che le variabili impostate e utilizzate durante la procedura non sono presenti in terminali diversi.

Per implementare il driver CSI per FSx for Lustre in un cluster Amazon EKS

1. Imposta alcune variabili da utilizzare nei passaggi rimanenti. Sostituiscilo *`my-csi-fsx-cluster`* con il nome del cluster di test che desideri creare e *`region-code`* con Regione AWS quello in cui desideri creare il cluster di test.

```
export cluster_name=my-csi-fsx-cluster
export region_code=region-code
```

2. Crea un cluster di test.

```
eksctl create cluster \
  --name $cluster_name \
  --region $region_code \
  --with-oidc \
  --ssh-access \
  --ssh-public-key my-key
```

Il provisioning del cluster richiede diversi minuti. Durante la creazione del cluster, vengono generate diverse righe di output. L'ultima riga di output è simile alla seguente riga di esempio.

```
[#]  EKS cluster "my-csi-fsx-cluster" in "region-code" region is ready
```

3. Crea un account Kubernetes di servizio per il driver e collega la policy `AmazonFSxFullAccess` AWS-managed all'account di servizio con il seguente comando. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

```
eksctl create iamserviceaccount \
  --name fsx-csi-controller-sa \
  --namespace kube-system \
  --cluster $cluster_name \
  --attach-policy-arn arn:aws:iam::aws:policy/AmazonFSxFullAccess \
  --approve \
  --role-name AmazonEKSFSxLustreCSIDriverFullAccess \
  --region $region_code
```

Vengono visualizzate diverse righe di output quando l'account del servizio viene creato. Le ultime righe dell'output sono simili a quelle riportate di seguito.

```
[#]  1 task: {
    2 sequential sub-tasks: {
        create IAM role for serviceaccount "kube-system/fsx-csi-controller-sa",
        create serviceaccount "kube-system/fsx-csi-controller-sa",
    } }
```

```
[#]  building iamserviceaccount stack "eksctl-my-csi-fsx-cluster-addon-
iamserviceaccount-kube-system-fsx-csi-controller-sa"
[#]  deploying stack "eksctl-my-csi-fsx-cluster-addon-iamserviceaccount-kube-
system-fsx-csi-controller-sa"
[#]  waiting for CloudFormation stack "eksctl-my-csi-fsx-cluster-addon-
iamserviceaccount-kube-system-fsx-csi-controller-sa"
[#]  created serviceaccount "kube-system/fsx-csi-controller-sa"
```

Annota il nome dello AWS CloudFormation stack che è stato distribuito. Nell'output di esempio precedente, lo stack è denominato `eksctl-my-csi-fsx-cluster-addon-iamserviceaccount-kube-system-fsx-csi-controller-sa`.

4. Implementare il driver mediante il comando seguente. Sostituisci *release-X.XX* con il ramo desiderato. Il ramo master non è supportato perché potrebbe contenere funzionalità prossime al rilascio che sono incompatibili con la versione stabile del driver attualmente rilasciata. Si consiglia di utilizzare l'ultima versione. Per un elenco delle filiali attive, vedi aws-fsx-csi-driversu. GitHub

> **Note**
>
> Puoi visualizzare il contenuto applicato in aws-fsx-csi-driver su GitHub.

```
kubectl apply -k "github.com/kubernetes-sigs/aws-fsx-csi-driver/deploy/kubernetes/
overlays/stable/?ref=release-X.XX"
```

Di seguito viene riportato un output di esempio:

```
serviceaccount/fsx-csi-controller-sa created
serviceaccount/fsx-csi-node-sa created
clusterrole.rbac.authorization.k8s.io/fsx-csi-external-provisioner-role created
clusterrole.rbac.authorization.k8s.io/fsx-external-resizer-role created
clusterrolebinding.rbac.authorization.k8s.io/fsx-csi-external-provisioner-binding
 created
clusterrolebinding.rbac.authorization.k8s.io/fsx-csi-resizer-binding created
deployment.apps/fsx-csi-controller created
daemonset.apps/fsx-csi-node created
csidriver.storage.k8s.io/fsx.csi.aws.com created
```

5. Annota l'ARN per il ruolo che è stato creato. Se non l'hai notato in precedenza e non l'hai più disponibile nell' AWS CLI output, puoi fare quanto segue per vederlo nel AWS Management Console.

   a. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.
   b. Assicurati che la console sia impostata sulla stessa in Regione AWS cui hai creato il tuo ruolo IAM, quindi seleziona Stacks.
   c. Selezionare la pila denominata `eksctl-`*`my-csi-fsx-cluster`*`-addon-iamserviceaccount-kube-system-`*`fsx-csi-controller-sa`*.
   d. Selezionare la scheda Outputs (Output). L'ARN Role1 è elencato nella pagina Output (1).

6. Applica una patch alla implementazione del driver per aggiungere l'account di servizio creato precedentemente con il seguente comando. Sostituisci l'ARN con l'ARN che hai annotato. Sostituisci *`111122223333`* con l'ID del tuo account. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:`

   ```
   kubectl annotate serviceaccount -n kube-system fsx-csi-controller-sa \
     eks.amazonaws.com/role-
   arn=arn:aws:iam::111122223333:role/AmazonEKSFSxLustreCSIDriverFullAccess --
   overwrite=true
   ```

   Di seguito viene riportato un output di esempio:

   ```
   serviceaccount/fsx-csi-controller-sa annotated
   ```

Per implementare una classe di storage Kubernetes, una dichiarazione di volume persistente e un'applicazione di esempio al fine di verificare il funzionamento del driver CSI

Questa procedura utilizza il driver CSI (Container Storage Interface) di FSx per Lustre disponibile nel repository GitHub per utilizzare un volume FSx per Lustre con provisioning dinamico.

1. Annota il gruppo di sicurezza per il cluster. Puoi vederlo nella AWS Management Console sezione Rete o usando il seguente AWS CLI comando.

   ```
   aws eks describe-cluster --name $cluster_name --query
    cluster.resourcesVpcConfig.clusterSecurityGroupId
   ```

2. Crea un gruppo di sicurezza per il file system Amazon FSx in base ai criteri indicati nella sezione Gruppi di sicurezza Amazon VPC della Guida per l'utente di Amazon FSx per Lustre. Per il VPC, seleziona il VPC del cluster come mostrato nella sezione Reti. Per i "gruppi di sicurezza associati ai client Lustre", usa il gruppo di sicurezza del cluster. Non specificare alcun valore nelle regole in uscita per consentire Tutto il traffico.
3. Scarica il manifesto della classe di storage con il comando seguente.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/
master/examples/kubernetes/dynamic_provisioning/specs/storageclass.yaml
```

4. Modifica la sezione dei parametri del file `storageclass.yaml`. Sostituisci ogni *example value* con i valori in tuo possesso.

```
parameters:
  subnetId: subnet-0eabfaa81fb22bcaf
  securityGroupIds: sg-068000ccf82dfba88
  deploymentType: PERSISTENT_1
  automaticBackupRetentionDays: "1"
  dailyAutomaticBackupStartTime: "00:00"
  copyTagsToBackups: "true"
  perUnitStorageThroughput: "200"
  dataCompressionType: "NONE"
  weeklyMaintenanceStartTime: "7:09:00"
  fileSystemTypeVersion: "2.12"
```

- **`subnetId`**: l'ID della sottorete in cui creare il file system Amazon FSx per Lustre. Amazon FSx for Lustre non è supportato in tutte le zone di disponibilità. Aprire la console Amazon FSx for Lustre all'indirizzo https://console.aws.amazon.com/fsx/ per confermare che la sottorete che si desidera utilizzare si trovi in una zona di disponibilità supportata. La sottorete può includere i nodi oppure può essere una sottorete o un VPC differente:
  - È possibile verificare la presenza di sottoreti di nodi in AWS Management Console selezionando il gruppo di nodi nella sezione Calcolo.
  - Se la sottorete specificata non è la stessa sottorete in cui sono presenti i nodi, i VPC devono essere connessi, pertanto è necessario verificare che le porte richieste siano aperte nei gruppi di sicurezza.
- **`securityGroupIds`**: l'ID del gruppo di sicurezza creato per il file system.

- **`deploymentType`**(facoltativo): il tipo di implementazione del file system. I valori validi sono `SCRATCH_1`, `SCRATCH_2`, `PERSISTENT_1` e `PERSISTENT_2`. Per ulteriori informazioni sui tipi di implementazione, consultare Creare il file system Amazon FSx for Lustre.
- altri parametri (opzionale): per informazioni sugli altri parametri, consulta Modifica su. StorageClass GitHub

5. Crea il manifesto della classe di storage.

```
kubectl apply -f storageclass.yaml
```

Di seguito viene riportato un output di esempio:

```
storageclass.storage.k8s.io/fsx-sc created
```

6. Scaricare il manifesto della dichiarazione di volume persistente.

```
curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/
master/examples/kubernetes/dynamic_provisioning/specs/claim.yaml
```

7. (Facoltativo) Modificare il file `claim.yaml`. Cambia *`1200Gi`* in uno dei valori di incremento riportati di seguito in base ai tuoi requisiti di archiviazione e al `deploymentType` selezionato in un passaggio precedente.

```
storage: 1200Gi
```

- `SCRATCH_2` e `PERSISTENT` – **`1.2 TiB`**, **`2.4 TiB`** o incrementi di 2,4 TiB su 2,4 TiB.
- `SCRATCH_1` – **`1.2 TiB`**, **`2.4 TiB`**, **`3.6 TiB`** o incrementi di 3,6 TiB su 3,6 TiB.

8. Creare la dichiarazione di volume persistente.

```
kubectl apply -f claim.yaml
```

Di seguito viene riportato un output di esempio:

```
persistentvolumeclaim/fsx-claim created
```

9. Verificare che il file system sia stato sottoposto a provisioning.

```
kubectl describe pvc
```

Di seguito viene riportato un output di esempio:

```
Name:          fsx-claim
Namespace:     default
StorageClass:  fsx-sc
Status:        Bound
[...]
```

> **Note**
>
> `Status` può apparire `Pending` per 5-10 minuti, prima di passare a `Bound`. Non continuare con il passo successivo fino a quando `Status` non è `Bound`. Se nel campo `Status` viene visualizzato `Pending` per più di 10 minuti, utilizza i messaggi di avviso negli `Events` come riferimento per risolvere eventuali problemi.

10. Implementare un'applicazione di esempio.

```
kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-fsx-csi-driver/master/examples/kubernetes/dynamic_provisioning/specs/pod.yaml
```

11. Verificare che l'applicazione di esempio sia in esecuzione.

```
kubectl get pods
```

Di seguito viene riportato un output di esempio:

```
NAME      READY   STATUS              RESTARTS   AGE
fsx-app   1/1     Running             0          8s
```

12. Verifica che l'applicazione abbia montato correttamente il file system.

```
kubectl exec -ti fsx-app -- df -h
```

Di seguito viene riportato un output di esempio:

```
Filesystem                Size  Used Avail Use% Mounted on
overlay                    80G  4.0G   77G   5% /
tmpfs                      64M     0   64M   0% /dev
tmpfs                     3.8G     0  3.8G   0% /sys/fs/cgroup
```

```
192.0.2.0@tcp:/abcdef01      1.1T  7.8M  1.1T   1% /data
/dev/nvme0n1p1                80G  4.0G   77G   5% /etc/hosts
shm                           64M     0   64M   0% /dev/shm
tmpfs                        6.9G   12K  6.9G   1% /run/secrets/kubernetes.io/
serviceaccount
tmpfs                        3.8G     0  3.8G   0% /proc/acpi
tmpfs                        3.8G     0  3.8G   0% /sys/firmware
```

13. Verificare che i dati siano stati scritti nel file system FSx for Lustre dall'app di esempio.

```
kubectl exec -it fsx-app -- ls /data
```

Di seguito viene riportato un output di esempio:

```
out.txt
```

Questo output esemplificativo mostra che l'app di esempio ha eseguito correttamente la scrittura del file `out.txt` nel file system.

> **Note**
>
> Prima di eliminare il cluster, assicurati di eliminare il file system FSx per Lustre. Per ulteriori informazioni, consulta Eliminazione di risorse nella Guida per l'utente di FSx per Lustre.

Driver CSI Amazon FSx per NetApp ONTAP

NetApp'sAstra Tridentfornisce un'orchestrazione dinamica dello storage utilizzando un driver conforme a Container Storage Interface (CSI). Ciò consente ai cluster Amazon EKS di gestire il ciclo di vita dei volumi persistenti (PV) supportati dai file system Amazon FSx for ONTAP. NetApp Per iniziare, consulta Usare Astra Trident con Amazon FSx NetApp for ONTAP nella documentazione. Astra Trident

Amazon FSx for NetApp ONTAP è un servizio di storage che consente di avviare ed eseguire ONTAP file system completamente gestiti nel cloud. ONTAPè una tecnologia di NetApp's file system che fornisce un set ampiamente adottato di funzionalità di accesso e gestione dei dati. FSx for ONTAP offre le funzionalità, le prestazioni e le API dei NetApp file system locali con l'agilità, la

scalabilità e la semplicità di un servizio completamente gestito. AWS Per ulteriori informazioni, consultare la Guida per l'utente di FSx per ONTAP.

Driver CSI per Amazon FSx per OpenZFS

Amazon FSx per OpenZFS è un servizio di archiviazione di file interamente gestito che semplifica il trasferimento di dati su AWS da ZFS on-premise o altri file server basati su Linux. Puoi farlo senza modificare il codice dell'applicazione o la modalità di gestione dei dati. Offre un'archiviazione di file altamente affidabile, scalabile, efficiente e ricca di funzionalità basata sul file system OpenZFS open source. Combina queste funzionalità con l'agilità, la scalabilità e la semplicità di un servizio AWS interamente gestito. Per ulteriori informazioni, consulta la Guida per l'utente di Amazon FSx per OpenZFS.

Il driver Container Storage Interface (CSI) di Amazon FSx per OpenZFS mette a disposizione un'interfaccia CSI che consente ai cluster Amazon EKS di gestire il ciclo di vita dei volumi Amazon FSx per OpenZFS. Per implementare il driver CSI di Amazon FSx per OpenZFS nel tuo cluster Amazon EKS, consulta `aws-fsx-openzfs-csi-driver` su GitHub.

Driver CSI di Amazon File Cache

Amazon File Cache è una cache ad alta velocità completamente gestita su AWS che viene utilizzata per elaborare i dati dei file indipendentemente da dove sono archiviati. Amazon File Cache carica automaticamente i dati nella cache quando vi si accede per la prima volta e li rilascia quando non vengono utilizzati. Per ulteriori informazioni, consulta la Guida per l'utente di Amazon File Cache.

Il driver Container Storage Interface (CSI) di Amazon File Cache mette a disposizione un'interfaccia CSI che consente ai cluster Amazon EKS di gestire il ciclo di vita delle cache di file di Amazon. Per implementare il driver CSI di Amazon File Cache nel tuo cluster Amazon EKS, consulta `aws-file-cache-csi-driver` su GitHub.

Driver CSI di Mountpoint per Amazon S3

Con il driver Mountpoint per Amazon S3 Container Storage Interface (CSI), Kubernetes le tue applicazioni possono accedere agli oggetti Amazon S3 tramite un'interfaccia di file system, ottenendo un throughput aggregato elevato senza modificare alcun codice applicativo. Basato su Mountpoint per Amazon S3, il driver CSI presenta un bucket Amazon S3 come volume, a cui è possibile

accedere tramite i container in Amazon EKS e i cluster Kubernetes autogestiti. Questo argomento illustra come implementare il driver CSI di Mountpoint per Amazon S3 nel cluster Amazon EKS.

Considerazioni

- Il driver CSI di Mountpoint per Amazon S3 non è compatibile con le immagini di container basate su Windows.
- Il driver CSI di Mountpoint per Amazon S3 non supporta AWS Fargate. Tuttavia, sono supportati i container in esecuzione in Amazon EC2 (con Amazon EKS o un'installazione Kubernetes personalizzata).
- Il driver CSI di Mountpoint per Amazon S3 supporta solo il provisioning statico. Non sono supportati il provisioning dinamico o la creazione di nuovi bucket.

> **Note**
>
> Il provisioning statico si riferisce all'utilizzo di un bucket Amazon S3 esistente specificato come `bucketName` `volumeAttributes` nell'oggetto. `PersistentVolume` Per ulteriori informazioni, consulta Provisioning statico su GitHub.

- I volumi montati con il driver CSI di Mountpoint per Amazon S3 non supportano tutte le funzionalità del file system POSIX. Per i dettagli sul comportamento del file system, consulta Comportamento del file system Mountpoint per Amazon S3 su GitHub.

Prerequisiti

- Un provider AWS Identity and Access Management (IAM) OpenID Connect (OIDC) esistente per il tuo cluster. Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- Versione 2.12.3 o successiva di quella AWS CLI installata e configurata sul dispositivo o. AWS CloudShell
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:

Creazione di una policy IAM

Il driver CSI di Mountpoint per Amazon S3 richiede le autorizzazioni Amazon S3 per interagire con il file system. In questa sezione viene descritto come creare una policy IAM che conceda le autorizzazioni necessarie.

La seguente policy di esempio segue i suggerimenti delle autorizzazioni IAM per Mountpoint. In alternativa, è possibile utilizzare la politica AWS gestita AmazonS3FullAccess, ma questa politica gestita concede più autorizzazioni di quelle per cui sono necessarie. Mountpoint

Per ulteriori informazioni sulle autorizzazioni consigliate per Mountpoint, consulta Autorizzazioni IAM per Mountpoint su GitHub.

Creare una policy IAM tramite la console IAM

1. Aprire la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Policy.
3. Nella pagina Policy, scegli Crea policy.
4. Per Editor di policy, scegli JSON.
5. In Editor di policy, copia e incolla quanto segue:

> **Important**
>
> Sostituisci `DOC-EXAMPLE-BUCKET1` con il nome del bucket Amazon S3.

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
            "Sid": "MountpointFullBucketAccess",
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1"
            ]
        },
```

```
        {
            "Sid": "MountpointFullObjectAccess",
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:AbortMultipartUpload",
                "s3:DeleteObject"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*"
            ]
        }
    ]
}
```

I bucket di directory, introdotti con la classe di storage Amazon S3 Express One Zone, utilizzano un meccanismo di autenticazione diverso rispetto ai bucket generici. Invece di usare `s3:*` le azioni, dovresti usare l'azione. `s3express:CreateSession` Per informazioni sui bucket di directory, consulta Directory buckets nella Amazon S3 User Guide.

Di seguito è riportato un esempio di politica di privilegi minimi da utilizzare per un bucket di directory.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "s3express:CreateSession",
            "Resource": "arn:aws:s3express:aws-region:111122223333:bucket/DOC-
EXAMPLE-BUCKET1--az_id--x-s3"
        }
    ]
}
```

6. Seleziona Successivo.
7. Nella pagina Verifica e crea, assegna un nome alla policy. Questa procedura guidata di esempio utilizza il nome `AmazonS3CSIDriverPolicy`.
8. Scegli Crea policy.

Creazione di un ruolo IAM

Il driver CSI di Mountpoint per Amazon S3 richiede le autorizzazioni Amazon S3 per interagire con il file system. In questa sezione viene descritto come creare un ruolo IAM per delegare queste autorizzazioni. Per creare questo ruolo, puoi utilizzare `eksctl`, la console IAM o la AWS CLI.

> **Note**
>
> La policy IAM `AmazonS3CSIDriverPolicy` è stata creata nella sezione precedente.

eksctl

Creazione del ruolo IAM del driver CSI di Mountpoint per Amazon S3 con **`eksctl`**

Per creare il ruolo IAM e l'account di servizio Kubernetes, esegui i comandi seguenti. Questi comandi collegano anche la policy IAM `AmazonS3CSIDriverPolicy` al ruolo, annotano l'account di servizio Kubernetes (`s3-csi-controller-sa`) con l'ARN (Amazon Resource Name) del ruolo IAM e aggiungono il nome dell'account di servizio Kubernetes alla policy di attendibilità per il ruolo IAM.

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
    --cluster $CLUSTER_NAME \
    --attach-policy-arn $POLICY_ARN \
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

IAM console

Per creare il tuo Mountpoint ruolo IAM per il driver CSI per Amazon S3 con AWS Management Console

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.

2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Nella pagina Ruoli, seleziona Crea ruolo.
4. Nella pagina Select trusted entity (Seleziona entità attendibile), esegui le operazioni seguenti:
   a. Nella sezione Tipo di identità attendibile, scegli Identità Web.
   b. Per Identity provider (Provider di identità), scegli l'URL del provider OpenID Connect per il cluster (come riportato in Overview [Panoramica] in Amazon EKS).

      Se non viene visualizzato alcun URL, consulta la sezione Prerequisiti.
   c. Per Pubblico, scegli `sts.amazonaws.com`.
   d. Seleziona Successivo.
5. Nella pagina Add permissions (Aggiungi autorizzazioni), esegui le operazioni seguenti:
   a. Nella casella Filtra policy, inserisci **`AmazonS3CSIDriverPolicy`**.

      > **Note**
      >
      > Questa policy è stata creata nella sezione precedente.

   b. Seleziona la casella di controllo a sinistra del risultato `AmazonS3CSIDriverPolicy` restituito dalla ricerca.
   c. Seleziona Successivo.
6. Nella pagina Name, review, and create (Assegna un nome, rivedi e crea), esegui le operazioni seguenti:
   a. Per Role name (Nome ruolo), inserisci un nome univoco per il ruolo, ad esempio **`AmazonEKS_S3_CSI_DriverRole`**.
   b. In Aggiungi tag (facoltativo), aggiungi metadati al ruolo collegando i tag come coppie chiave-valore. Per ulteriori informazioni sull'utilizzo di tag in IAM, consulta la sezione Applicazione di tag alle risorse IAM nella Guida per l'utente di IAM.
   c. Scegli Crea ruolo.
7. Dopo aver creato il ruolo, sceglilo nella console in modo da aprirlo per la modifica.
8. Scegli la scheda Trust relationships (Relazioni di attendibilità), quindi scegli Edit trust policy (Modifica policy di attendibilità).
9. Cercare il risultato finale simile al seguente:

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud":
 "sts.amazonaws.com"
```

Inserisci una virgola alla fine della riga precedente, quindi aggiungine una seguente dopo quella riga. *region-code*Sostituiscilo con quello in Regione AWS cui si trova il tuo cluster. Sostituisci *EXAMPLED539D4633E53DE1B71EXAMPLE* con il gestore di ID OIDC del tuo cluster.

```
"oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub":
 "system:serviceaccount:kube-system:s3-csi-*"
```

10. Modifica l'operatore `Condition` da `"StringEquals"` a `"StringLike"`.
11. Scegli Aggiorna policy per concludere.

AWS CLI

Per creare il tuo Mountpoint ruolo IAM per il driver CSI per Amazon S3 con AWS CLI

1. Visualizza l'URL del provider OIDC del cluster. Sostituisci *my-cluster* con il nome del cluster. Se l'output dal comando è `None`, rivedi i Prerequisiti.

```
aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text
```

Di seguito viene riportato un output di esempio:

```
https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
```

2. Crea il ruolo IAM, concedendo all'account di servizio Kubernetes l'operazione `AssumeRoleWithWebIdentity`.

   a. Copia i contenuti seguenti in un file denominato *aws-s3-csi-driver-trust-policy*`.json`. Sostituisci *111122223333* con l'ID del tuo account. Sostituire *EXAMPLED539D4633E53DE1B71EXAMPLE* e *region-code* con i valori restituiti nella fase precedente.

```
{
  "Version": "2012-10-17",
```

```
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
      },
      "Action": "sts:AssumeRoleWithWebIdentity",
      "Condition": {
        "StringLike": {
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:s3-csi-*",
          "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com"
        }
      }
    }
  ]
}
```

b. Creare il ruolo. È possibile modificare *AmazonEKS_S3_CSI_DriverRole* con un nome diverso. In tal caso, assicurati di modificarlo anche nelle fasi successive.

```
aws iam create-role \
  --role-name AmazonEKS_S3_CSI_DriverRole \
  --assume-role-policy-document file://"aws-s3-csi-driver-trust-policy.json"
```

3. Collega la policy IAM creata in precedenza al ruolo con il seguente comando.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonS3CSIDriverPolicy \
  --role-name AmazonEKS_S3_CSI_DriverRole
```

> **Note**
>
> La policy IAM `AmazonS3CSIDriverPolicy` è stata creata nella sezione precedente.

4. Salta questa fase se stai installando il driver come componente aggiuntivo di Amazon EKS. Per le installazioni autogestite del driver, crea Kubernetes account di servizio annotati con l'ARN del ruolo IAM che hai creato.

   a. Salvare i seguenti contenuti in un file denominato *mountpoint-s3-service-account*`.yaml`. Sostituisci *111122223333* con l'ID del tuo account.

   ```
   ---
   apiVersion: v1
   kind: ServiceAccount
   metadata:
     labels:
       app.kubernetes.io/name: aws-mountpoint-s3-csi-driver
     name: mountpoint-s3-csi-controller-sa
     namespace: kube-system
     annotations:
       eks.amazonaws.com/role-arn:
    arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
   ```

   b. Crea l'account di servizio Kubernetes sul cluster. L'account del servizio Kubernetes (`mountpoint-s3-csi-controller-sa`) è annotato con il ruolo IAM creato precedentemente e denominato *AmazonEKS_S3_CSI_DriverRole*.

   ```
   kubectl apply -f mountpoint-s3-service-account.yaml
   ```

   > **Note**
   >
   > Quando implementi il plugin tramite questa procedura, il plugin crea ed è configurato per l'utilizzo di un account di servizio denominato `s3-csi-driver-sa`.

Installazione del driver CSI di Mountpoint per Amazon S3

Puoi installare il driver CSI di Mountpoint per Amazon S3 tramite il componente aggiuntivo di Amazon EKS. Puoi usare`eksctl`, the AWS Management Console, o the AWS CLI per aggiungere il componente aggiuntivo al tuo cluster.

Facoltativamente, puoi installare il driver CSI Mountpoint per Amazon S3 come installazione autogestita. Per istruzioni su come eseguire un'installazione autogestita, consulta Installazione su GitHub.

eksctl

Per aggiungere il componente aggiuntivo Amazon S3 CSI utilizzando **`eksctl`**

Esegui il comando seguente. Sostituisci *`my-cluster`* con il nome del cluster, *`111122223333`* con il tuo ID account e *`AmazonEKS_S3_CSI_DriverRole`* con il nome del ruolo IAM creato in precedenza.

```
eksctl create addon --name aws-mountpoint-s3-csi-driver --cluster my-cluster --
service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
 --force
```

Se si rimuove l'opzione **`--force`** e tutte le impostazioni del componente aggiuntivo Amazon EKS sono in conflitto con le impostazioni esistenti, l'aggiornamento del componente aggiuntivo Amazon EKS non avrà successo. Verrà quindi visualizzato un messaggio di errore con la procedura utile a risolvere il conflitto. Prima di specificare questa opzione, assicurati che il componente aggiuntivo di Amazon EKS non gestisca le impostazioni che devi gestire tu, perché questa opzione le sovrascrive. Per ulteriori informazioni su altre opzioni per questa impostazione, consulta Componenti aggiuntivi nella documentazione di `eksctl`. Per ulteriori informazioni sulla gestione dei campi Kubernetes di Amazon EKS, consulta Gestione dei campi Kubernetes.

AWS Management Console

Per aggiungere il componente Mountpoint aggiuntivo CSI per Amazon S3 utilizzando AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, seleziona Cluster.
3. Scegli il nome del cluster per cui desideri configurare il componente aggiuntivo CSI Mountpoint per Amazon S3.
4. Seleziona la scheda Componenti aggiuntivi.
5. Scegli Ottieni altri componenti aggiuntivi.
6. Nella pagina Seleziona componenti aggiuntivi, procedi come segue:

a. Nella sezione Amazon EKS-Addons, seleziona la casella di controllo per Amazon MountpointS3 CSI Driver.

b. Seleziona Successivo.

7. Nella pagina Configura le impostazioni dei componenti aggiuntivi selezionati, procedi come segue:

a. Seleziona la Versione che desideri utilizzare.

b. Per Select IAM role, seleziona il nome di un ruolo IAM a cui hai collegato la policy IAM del driver CSI Mountpoint per Amazon S3.

c. (Facoltativo) È possibile espandere le Impostazioni di configurazione facoltative. Se selezioni Sostituisci per Metodo di risoluzione dei conflitti, una o più impostazioni per il componente aggiuntivo esistente possono essere sovrascritte con le impostazioni del componente aggiuntivo di Amazon EKS. Se rimuovi questa opzione e c'è un conflitto con le impostazioni esistenti, l'operazione non va a buon fine e viene visualizzato un messaggio di errore per aiutarti a risolvere il conflitto. Prima di selezionare questa opzione, assicurarsi che il componente aggiuntivo Amazon EKS non gestisca le impostazioni che devono essere gestite dall'utente.

d. Seleziona Successivo.

8. Nella pagina Rivedi e aggiungi, scegli Crea. Una volta completata l'installazione, viene visualizzato il componente aggiuntivo.

AWS CLI

Per aggiungere il componente Mountpoint aggiuntivo CSI per Amazon S3 utilizzando AWS CLI

Esegui il comando seguente. Sostituire *my-cluster* con il nome del cluster, *111122223333* con il proprio ID account e *AmazonEKS_S3_CSI_DriverRole* con il nome del ruolo creato in precedenza.

```
aws eks create-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-
driver \
  --service-account-role-arn
 arn:aws:iam::111122223333:role/AmazonEKS_S3_CSI_DriverRole
```

Configurazione di Mountpoint per Amazon S3

Nella maggior parte dei casi, puoi configurare Mountpoint per Amazon S3 solo con un nome di bucket. Per istruzioni sulla configurazione di Mountpoint per Amazon S3, consulta Configurazione di Mountpoint per Amazon S3 su GitHub.

Implementazione di un'applicazione di esempio

Puoi implementare il provisioning statico al driver su un bucket Amazon S3 esistente. Per ulteriori informazioni, consulta Provisioning statico su GitHub.

Rimozione del driver CSI Mountpoint per Amazon S3

Sono possibili due opzioni per rimuovere un componente aggiuntivo Amazon EKS.

- Mantenere il software aggiuntivo sul cluster – Questa opzione rimuove la gestione di Amazon EKS su qualsiasi impostazione. Inoltre, rimuove la possibilità per Amazon EKS di notificare gli aggiornamenti e aggiornare automaticamente il componente aggiuntivo Amazon EKS dopo l'avvio di un aggiornamento. Tuttavia, mantiene il software aggiuntivo sul cluster. Questa opzione rende il componente aggiuntivo come autogestito anziché come un componente aggiuntivo Amazon EKS. Con questa opzione, non ci sono tempi di inattività per il componente aggiuntivo. I comandi in questa procedura utilizzano questa opzione.
- Rimuovere completamente il software aggiuntivo dal cluster – Consigliamo di rimuovere il componente aggiuntivo Amazon EKS dal cluster solo se non ci sono risorse nel cluster che dipendono dallo stesso. Per eseguire questa operazione, elimina `--preserve` dal comando utilizzato in questa procedura.

Se al componente aggiuntivo è associato un account IAM, l'account IAM non viene rimosso.

Puoi usare`eksctl`, il AWS Management Console, o the AWS CLI per rimuovere il componente aggiuntivo Amazon S3 CSI.

eksctl

Per rimuovere il componente aggiuntivo Amazon S3 CSI utilizzando **`eksctl`**

Sostituire *`my-cluster`* con il nome del cluster ed eseguire il comando seguente.

```
eksctl delete addon --cluster my-cluster --name aws-mountpoint-s3-csi-driver --
preserve
```

AWS Management Console

Per rimuovere il componente aggiuntivo Amazon S3 CSI utilizzando AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, seleziona Cluster.
3. Scegli il nome del cluster per cui desideri rimuovere il componente aggiuntivo CSI per Amazon EBS.
4. Seleziona la scheda Componenti aggiuntivi.
5. Scegli Mountpointil driver Amazon S3 CSI.
6. Scegli Rimuovi.
7. Nella finestra di dialogo di conferma Rimuovi: aws-mountpoint-s 3-csi driver, procedi come segue:
   a. Se desideri che Amazon EKS interrompa la gestione delle impostazioni per il componente aggiuntivo, seleziona Mantieni sul cluster. Effettuare questa operazione se si desidera conservare il software aggiuntivo sul cluster. In questo modo sarà possibile gestire autonomamente tutte le impostazioni del componente aggiuntivo.
   b. Specificare **`aws-mountpoint-s3-csi-driver`**.
   c. Selezionare Rimuovi.

AWS CLI

Per rimuovere il componente aggiuntivo Amazon S3 CSI utilizzando AWS CLI

Sostituire *`my-cluster`* con il nome del cluster ed eseguire il comando seguente.

```
aws eks delete-addon --cluster-name my-cluster --addon-name aws-mountpoint-s3-csi-
driver --preserve
```

Controller di snapshot CSI

Il controller di snapshot CSI (Container Storage Interface) consente l'uso della funzionalità di snapshotting nei driver CSI compatibili, come il driver CSI di Amazon EBS.

Di seguito sono elencati alcuni punti da considerare quando si utilizza il controller di snapshot CSI.

- Il controller di snapshot deve essere installato insieme a un driver CSI con funzionalità di snapshotting. Il driver CSI di Amazon EBS supporta la creazione di snapshot Amazon EBS di volumi gestiti CSI di Amazon EBS. Per le istruzioni di installazione, consulta Driver CSI per Amazon EBS.
- Kubernetes non supporta snapshot di volumi serviti tramite migrazione CSI, come i volumi Amazon EBS che utilizzano una `StorageClass` con provisioner `kubernetes.io/aws-ebs`. I volumi devono essere creati con una `StorageClass` che fa riferimento al provisioner del driver CSI, `ebs.csi.aws.com`. Per ulteriori informazioni sulla migrazione CSI, consulta Domande frequenti sulla migrazione CSI di Amazon EBS.

Consigliamo di installare il controller di snapshot CSI attraverso il componente aggiuntivo gestito di Amazon EKS. Per aggiungere un componente aggiuntivo di Amazon EKS al cluster, consulta Creazione di un componente aggiuntivo. Per ulteriori informazioni sui componenti aggiuntivi, consulta Componenti aggiuntivi Amazon EKS.

In alternativa, se desideri un'installazione autogestita del controller di snapshot CSI di Amazon EBS, consulta Utilizzo nel Kubernetes upstream `external-snapshotter` su GitHub.

Reti Amazon EKS

Il cluster Amazon EKS viene creato in un VPC. Le reti pod sono fornite dal plugin CNI (Container Network Interface) di Amazon VPC. Per ulteriori informazioni sulle reti disponibili per il cluster, puoi consultare gli argomenti seguenti.

Argomenti

Requisiti e considerazioni su VPC e sottoreti di Amazon EKS

Durante la creazione di un cluster, in genere si specificano un VPC e due sottoreti che si trovano in zone di disponibilità diverse. Questo argomento offre una panoramica dei requisiti e delle considerazioni specifiche di Amazon EKS per il VPC e le sottoreti utilizzate con il cluster. Se non disponi di un VPC da utilizzare con Amazon EKS, puoi crearne uno utilizzando un modello fornito AWS CloudFormation da Amazon EKS. Se stai creando un cluster locale o esteso su AWS Outposts, consulta Requisiti e considerazioni su VPC e sottoreti del cluster locale Amazon EKS invece di questo argomento.

Considerazioni e requisiti relativi al VPC

Durante la creazione di un cluster, il VPC specificato deve soddisfare i requisiti e le considerazioni seguenti:

- Il VPC deve disporre di un numero sufficiente di indirizzi IP da mettere a disposizione per il cluster, i nodi e le altre risorse Kubernetes da creare. In caso contrario, è possibile provare ad aumentare il numero di indirizzi IP disponibili.

  È possibile farlo aggiornando la configurazione del cluster per modificare le sottoreti e i gruppi di sicurezza utilizzati dal cluster. Puoi eseguire l' AWS Management Console aggiornamento dalla versione più recente di e dalla AWS CLI`eksctl` versione `v0.164.0-rc.0` o successiva. AWS

CloudFormation Potrebbe essere necessario eseguire questa operazione per fornire alle sottoreti più indirizzi IP disponibili per aggiornare in modo corretto una versione del cluster.

> **Important**
>
> Tutte le sottoreti aggiunte devono trovarsi nello stesso insieme di AZ fornito in origine quando hai creato il cluster. Le nuove sottoreti devono soddisfare tutti gli altri requisiti, ad esempio devono disporre di sufficienti indirizzi IP.
> Supponiamo ad esempio che hai creato un cluster e specificato quattro sottoreti. Nell'ordine in cui le hai specificate, la prima sottorete si trova nella zona di disponibilità `us-west-2a`, la seconda e la terza si trovano nella zona di disponibilità `us-west-2b` e la quarta si trova nella zona di disponibilità `us-west-2c`. Se desideri modificare le sottoreti, devi fornire almeno una sottorete in ciascuna delle tre zone di disponibilità e le sottoreti devono trovarsi nello stesso VPC delle sottoreti originali.

Se hai bisogno di più indirizzi IP rispetto ai blocchi CIDR nel VPC, puoi aggiungere altri blocchi CIDR associando ulteriori blocchi di routing interdominio senza classi (CIDR) al VPC. L'associazione di blocchi CIDR privati (RFC 1918) e pubblici (non RFC 1918) può avvenire prima o dopo la creazione del cluster. Il riconoscimento del blocco CIDR associato a un VPC da parte del cluster può richiedere fino a cinque ore.

È possibile conservare l'utilizzo degli indirizzi IP tramite un gateway di transito con un VPC di servizi condivisi. Per ulteriori informazioni, consulta VPC isolati con servizi condivisi e Modelli di conservazione degli indirizzi IP instradabili di Amazon EKS VPC in una rete ibrida.

- Se vuoi che Kubernetes assegni indirizzi `IPv6` a Pods e servizi, associa un blocco CIDR `IPv6` al VPC. Per ulteriori informazioni, consulta Associazione di un blocco CIDR `IPv6` al VPC nella Guida per l'utente di Amazon VPC.
- Il VPC deve supportare il nome host `DNS` e la risoluzione `DNS`. In caso contrario, i nodi non possono registrarsi al cluster. Per ulteriori informazioni, consulta Attributi DNS nel VPC nella Guida per l'utente di Amazon VPC.
- Il VPC potrebbe richiedere l'utilizzo di endpoint VPC. AWS PrivateLink Per ulteriori informazioni, consulta Considerazioni e requisiti relativi alle sottoreti.

Se il cluster è stato creato con Kubernetes `1.14` o versioni precedenti, Amazon EKS ha aggiunto il tag seguente al VPC:

Chiave	Valore
`kubernetes.io/cluster/ my-cluster`	`owned`

Questo tag è stato utilizzato solo da Amazon EKS, per cui è possibile rimuoverlo senza influire sui servizi. Non è necessario con la versione `1.15` del cluster o con versioni successive.

Considerazioni e requisiti relativi alle sottoreti

Durante la creazione di un cluster, Amazon EKS crea da 2 a 4 interfacce di rete elastiche nelle sottoreti specificate. Queste interfacce di rete consentono la comunicazione tra il cluster e il VPC. Inoltre, abilitano alcune funzionalità di Kubernetes come `kubectl exec` e `kubectl logs`. Ogni interfaccia di rete creata da Amazon EKS ha il testo `Amazon EKS cluster-name` nella sua descrizione.

Amazon EKS può creare le proprie interfacce di rete in qualsiasi sottorete specificata al momento della creazione di un cluster. Puoi modificare le sottoreti in cui Amazon EKS crea le interfacce di rete dopo la creazione del cluster. Quando aggiorni la versione Kubernetes di un cluster, Amazon EKS elimina le interfacce di rete originali e ne crea di nuove. Queste interfacce di rete potrebbero essere create all'interno delle stesse sottoreti o in sottoreti diverse rispetto alle interfacce di rete originali. Per verificare in quali sottoreti vengono create le interfacce di rete, puoi limitare il numero di sottoreti specificate a due quando crei un cluster o aggiornare le sottoreti dopo aver creato il cluster.

Requisiti relativi alla sottorete per i cluster

Le sottoreti specificate al momento della creazione o dell'aggiornamento di un cluster devono soddisfare i seguenti requisiti:

- Ciascuna sottorete deve disporre di almeno sei indirizzi IP per l'uso da parte di Amazon EKS. Tuttavia, consigliamo di utilizzare almeno 16 indirizzi IP.
- Le sottoreti non possono risiedere in o in AWS Outposts una zona locale. AWS Wavelength AWS Tuttavia, se presenti nel VPC, puoi implementare nodi autogestiti e risorse Kubernetes in questi tipi di sottoreti.
- Le sottoreti possono essere pubbliche o private. Tuttavia, si consiglia di specificare sottoreti private, se possibile. Una sottorete pubblica è una sottorete con una tabella di instradamento che include un instradamento a un gateway Internet, mentre la sottorete privata non comprende tale instradamento.

- Le sottoreti non possono risiedere nelle seguenti zone di disponibilità:

Regione AWS	Nome Regione	ID della zona di disponibilità non consentita
us-east-1	Stati Uniti orientali (Virginia settentrionale)	use1-az3
us-west-1	Stati Uniti occidentali (California settentrionale)	usw1-az2
ca-central-1	Canada (Centrale)	cac1-az3

Utilizzo della famiglia di indirizzi IP per componente

La tabella seguente contiene la famiglia di indirizzi IP utilizzata da ogni componente di Amazon EKS. Puoi utilizzare un NAT (Network Address Translation) o un altro sistema di compatibilità per connetterti a questi componenti da indirizzi IP di origine in famiglie con il "No" valore per una voce di tabella.

La funzionalità può variare a seconda dell'impostazione IP family (`ipFamily`) del cluster. Questa impostazione modifica il tipo di indirizzi IP utilizzati per il CIDR blocco Kubernetes assegnato aServices. Un cluster con il valore di impostazione di IPv4 viene definito come unIPv4 cluster, mentre un cluster con il valore di impostazione di IPv6 viene definito come unIPv6 cluster.

Componente	**IPv4**solo indirizzi	**IPv6**solo indirizzi	indirizzi dual stack
Endpoint pubblico dell'API EKS	Sì	No	No
Endpoint VPC API EKS	Sì	No	No
Endpoint pubblico dell'API EKS Auth	Sì [1]	Sì [1]	Sì [1]
Endpoint VPC dell'API di autenticazione EKS	Sì [1]	Sì [1]	Sì [1]

Componente	**IPv4**solo indirizzi	**IPv6**solo indirizzi	indirizzi dual stack
Endpoint pubblico del cluster EKS	Sì	No	No
Endpoint privato del cluster EKS	Sì [2]	Sì [2]	No
Sottoreti del cluster EKS	Sì [2]	No	Sì [2]
Indirizzi IP primari del nodo	Sì [2]	No	Sì [2]
CIDRIntervallo di cluster per indirizzi Service IP	Sì [2]	Sì [2]	No
PodIndirizzi IP dal VPC CNI	Sì [2]	Sì [2]	No

Note

[1] L'endpoint è a doppio stack con entrambi gli indirizzi. `IPv4` `IPv6` Le applicazioni esterne AWS, i nodi del cluster e i pod all'interno del cluster possono raggiungere questo endpoint tramite l'uno o l'altro. `IPv4` `IPv6`

[2] È possibile scegliere tra un `IPv4` cluster e un `IPv6` cluster nell'impostazione IP family (`ipFamily`) del cluster quando si crea un cluster e questa impostazione non può essere modificata. È invece necessario scegliere un'impostazione diversa quando si crea un altro cluster e si migrano i carichi di lavoro.

Requisiti relativi alla sottorete per i nodi

Puoi implementare nodi e risorse Kubernetes nelle stesse sottoreti specificate al momento della creazione del cluster. Tuttavia, l'implementazione non è necessaria. Questo perché puoi implementare nodi e risorse Kubernetes anche in sottoreti che non hai specificato durante la creazione del cluster. L'implementazione di nodi in sottoreti diverse non comporta la creazione

di interfacce di rete cluster in tali sottoreti da parte di Amazon EKS. Qualsiasi sottorete in cui si implementano nodi e risorse Kubernetes deve soddisfare i seguenti requisiti:

- Per implementare tutti i nodi e le risorse Kubernetes, le sottoreti devono disporre di un numero sufficiente di indirizzi IP.
- Se desideri che Kubernetes assegni indirizzi `IPv6` a Pods e servizi, dovrai disporre di un blocco CIDR `IPv6` e di un blocco CIDR `IPv4` associati alla sottorete. Per ulteriori informazioni, consulta Associazione di un blocco CIDR `IPv6` alla sottorete nella Guida per l'utente di Amazon VPC. Le tabelle di instradamento associate alle sottoreti devono includere instradamenti a indirizzi `IPv4` e `IPv6`. Per maggiori informazioni, consulta Routes (Instradamenti) nella Guida dell'utente di Amazon VPC. Ai pod viene assegnato solo un indirizzo `IPv6`, mentre alle interfacce di rete create da Amazon EKS per il cluster e ai nodi vengono assegnati un indirizzo `IPv4` e uno `IPv6`.
- Se è necessario garantire ai Pods un accesso in entrata da Internet, assicurati di disporre di almeno una sottorete pubblica con un numero sufficiente di indirizzi IP disponibili su cui implementare i load balancer e gli ingressi. Puoi implementare sistemi di bilanciamento del carico nelle sottoreti pubbliche. I sistemi di bilanciamento del carico possono bilanciare il carico sui Pods in sottoreti private o pubbliche. Consigliamo di implementare i nodi su sottoreti private, se possibile.
- Se si prevede di implementare i nodi in una sottorete pubblica, quest'ultima deve essere configurata per assegnare automaticamente indirizzi `IPv4` pubblici o indirizzi `IPv6`. Se si implementano nodi in una sottorete privata a cui è associato un blocco CIDR `IPv6`, anche questa sottorete deve essere configurata per assegnare automaticamente indirizzi `IPv6`. Se hai utilizzato un AWS CloudFormation modello Amazon EKS per distribuire il tuo VPC dopo il 26 marzo 2020, questa impostazione è abilitata. Se hai utilizzato i modelli per implementare il VPC prima di tale data o utilizzi un VPC personale, devi abilitare questa impostazione manualmente. Per ulteriori informazioni, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv4` pubblici per la sottorete e Modifica dell'attributo di assegnazione degli indirizzi `IPv6` per la sottorete nella Guida per l'utente di Amazon VPC.
- Se la sottorete in cui si implementa un nodo è privata e la relativa tabella di instradamento non include un instradamento verso un dispositivo Network address translation (NAT) (`IPv4`) o un gateway egress-only (`IPv6`), aggiungere endpoint VPC utilizzando AWS PrivateLink al VPC. Gli endpoint VPC sono necessari per tutti i Servizi AWS nodi con Pods cui devono comunicare. Alcuni esempi includono Amazon ECR, Elastic Load Balancing, Amazon e CloudWatch Amazon AWS Security Token Service Simple Storage Service (Amazon S3). L'endpoint deve includere la sottorete in cui si trovano i nodi. Non tutti Servizi AWS supportano gli endpoint VPC. Per ulteriori informazioni, consulta Cos'è? AWS PrivateLink e AWS servizi che si integrano con AWS PrivateLink. Per un elenco di altri requisiti Amazon EKS, consulta Requisiti dei cluster privati.

- Se si desidera implementare i load balancer in una sottorete, quest'ultima deve presentare il tag seguente:
  - Sottoreti private

Chiave	Valore
`kubernetes.io/role/internal-elb`	`1`

  - Sottoreti pubbliche

Chiave	Valore
`kubernetes.io/role/elb`	`1`

In passato, quando si creava un cluster Kubernetes versione `1.18` e precedenti, Amazon EKS aggiungeva il seguente tag a tutte le sottoreti specificate.

Chiave	Valore
`kubernetes.io/cluster/` *`my-cluster`*	`shared`

Adesso, se crei un nuovo cluster Kubernetes, Amazon EKS non aggiunge alcun tag alle sottoreti. Se il tag era applicato a sottoreti utilizzate da un cluster che in precedenza era una versione precedente a `1.19`, il tag non veniva rimosso automaticamente dalle sottoreti quando il cluster veniva aggiornato a una versione più recente. La versione `2.1.1` o precedenti di AWS Load Balancer Controller richiede questo tag. Se si utilizza una versione più recente di Load Balancer Controller, è possibile rimuovere il tag senza interrompere i servizi.

Se hai distribuito un VPC `eksctl`utilizzando o uno qualsiasi dei modelli AWS CloudFormation VPC di Amazon EKS, si applica quanto segue:

- In data 26 marzo 2020 o successiva. Gli indirizzi `IPv4` pubblici vengono assegnati automaticamente dalle sottoreti pubbliche ai nuovi nodi implementati nelle sottoreti pubbliche.
- Prima del 26 marzo 2020. Gli indirizzi `IPv4` pubblici non vengono assegnati automaticamente dalle sottoreti pubbliche ai nuovi nodi implementati nelle sottoreti pubbliche.

Questa modifica influisce sui nuovi gruppi di nodi implementati nelle sottoreti pubbliche nei modi seguenti:

- Gruppi di nodi gestiti. Se il gruppo di nodi viene implementato in una sottorete pubblica in data 22 aprile 2020 o successiva, la sottorete pubblica deve disporre dell'assegnazione automatica degli indirizzi IP pubblici abilitati. Per ulteriori informazioni, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv4` pubblici per la sottorete.
- Gruppi di nodi autogestiti Linux, Windows o Arm: se il gruppo di nodi viene implementato in una sottorete pubblica dal 26 marzo 2020 in poi, la sottorete pubblica dovà disporre dell'assegnazione automatica degli indirizzi IP pubblici. Altrimenti, i nodi devono essere avviati con un indirizzo IP pubblico. Per ulteriori informazioni, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv4` pubblici per la sottorete o Assegnazione di un indirizzo `IPv4` pubblico durante l'avvio dell'istanza.

Considerazioni e requisiti relativi alle sottoreti condivisi

È possibile utilizzare la Condivisione VPC per condividere sottoreti con altri account AWS all'interno della stessa AWS Organizations. È possibile creare cluster Amazon EKS in sottoreti condivise, tenendo a mente le seguenti considerazioni:

- Il proprietario della sottorete VPC deve condividere una sottorete con un account partecipante prima che tale account possa creare un cluster Amazon EKS al suo interno.
- Non è possibile avviare le risorse utilizzando il gruppo di sicurezza predefinito per il VPC, in quanto questo appartiene al proprietario. Inoltre, i partecipanti non possono avviare le risorse utilizzando i gruppi di sicurezza di proprietà di altri partecipanti o del proprietario.
- In una sottorete condivisa, il partecipante e il proprietario controllano separatamente i gruppi di sicurezza all'interno di ciascun account. Il proprietario della sottorete può visualizzare i gruppi di sicurezza che sono stati creati dai partecipanti, ma non può eseguire operazioni sugli stessi. Se il proprietario della sottorete desidera rimuovere o modificare questi gruppi di sicurezza, è il partecipante che ha creato il gruppo di sicurezza che deve eseguire l'operazione.
- Se un cluster viene creato da un partecipante, valgono le seguenti considerazioni:
  - Il ruolo IAM del cluster e i ruoli IAM del nodo devono essere creati in quell'account. Per ulteriori informazioni, consulta Ruolo IAM del cluster Amazon EKS e Ruolo IAM del nodo Amazon EKS.
  - Tutti i nodi devono essere creati dallo stesso partecipante, inclusi i gruppi di nodi gestiti.
- Il proprietario del VPC condiviso non può visualizzare, aggiornare o eliminare un cluster creato da un partecipante nella sottorete condivisa. Ciò si aggiunge alle risorse VPC alle quali ogni

account ha un accesso diverso. Per ulteriori informazioni, consulta la sezione Responsabilità e autorizzazioni per proprietari e partecipanti nella Guida per l'utente di Amazon VPC.

- Se si utilizza la funzionalità Rete personalizzata del Amazon VPC CNI plugin for Kubernetes, è necessario utilizzare le mappature degli ID delle zone di disponibilità elencate nell'account del proprietario per creare ciascuna `ENIConfig`. Per ulteriori informazioni, consulta Rete personalizzata per i pod.

Per ulteriori informazioni sulla condivisione delle sottoreti VPC, consulta la pagina Condivisione del VPC con altri account nella Guida per l'utente di Amazon VPC.

Creazione di un VPC per il cluster Amazon EKS

Puoi usare Amazon Virtual Private Cloud (Amazon VPC) per lanciare AWS risorse in una rete virtuale che hai definito. Questa rete virtuale è simile a una rete tradizionale da gestire all'interno del proprio data center, ma con i vantaggi dell'infrastruttura scalabile di Amazon Web Services. Prima di implementare cluster Amazon EKS di produzione, è preferibile approfondire le nozioni relative al servizio Amazon VPC. Per ulteriori informazioni, consulta la Guida utente Amazon VPC.

Un cluster Amazon EKS, i nodi e le risorse Kubernetes vengono implementati su un VPC. Se desideri utilizzare un VPC esistente con Amazon EKS, tale VPC dovrà soddisfare i requisiti descritti nella sezione Requisiti e considerazioni su VPC e sottoreti di Amazon EKS. Questo argomento descrive come creare un VPC che soddisfi i requisiti di Amazon EKS utilizzando un modello fornito AWS CloudFormation da Amazon EKS. Dopo l'implementazione di un modello, puoi visualizzare le risorse create dal modello per sapere esattamente quali risorse ha creato e la configurazione di tali risorse.

Prerequisito

Per creare un VPC destinato ad Amazon EKS, devi disporre delle autorizzazioni IAM necessarie alla creazione delle risorse per Amazon VPC. Queste risorse includono VPC, sottoreti, gruppi di sicurezza, tabelle di instradamento, instradamenti e gateway sia Internet che NAT. Per ulteriori informazioni, consulta la sezione Creare un VPC con una sottorete pubblica nella Guida per l'utente di Amazon VPC e l'elenco completo di Operazioni, risorse e chiavi di condizione per Amazon EC2 disponibile nella Guida di riferimento per l'autorizzazione del servizio.

È possibile creare un VPC con sottoreti sia pubbliche che private, solo pubbliche o solo private.

Public and private subnets

Questo VPC dispone di due sottoreti pubbliche e due private. Alle sottoreti pubbliche è associata una tabella di instradamento con un instradamento a un gateway Internet, Tuttavia, la tabella di instradamento di una sottorete privata non ha una route a un gateway Internet. Sia la sottorete pubblica che la sottorete privata vengono distribuite nella stessa zona di disponibilità. Le altre sottoreti pubbliche e private vengono implementate in una seconda zona di disponibilità nella stessa Regione AWS. Si consiglia questa opzione per la maggior parte delle implementazioni,

dal momento che consente di distribuire i nodi in sottoreti private. Questa opzione consente a Kubernetes di implementare i load balancer nelle sottoreti pubbliche in grado di bilanciare il carico del traffico nei Pods in esecuzione sui nodi nelle sottoreti private. Gli indirizzi `IPv4` pubblici vengono assegnati automaticamente ai nodi implementati nelle sottoreti pubbliche, tuttavia gli indirizzi `IPv4` pubblici non vengono assegnati ai nodi implementati nelle sottoreti private.

È anche possibile assegnare indirizzi `IPv6` ai nodi nelle sottoreti pubbliche e private. I nodi nelle sottoreti private possono comunicare con il cluster e altri Servizi AWS. I Pods possono comunicare con Internet tramite un gateway NAT utilizzando gli indirizzi `IPv4` o tramite un gateway Internet in uscita utilizzando gli indirizzi `IPv6` implementati in ciascuna zona di disponibilità. Viene implementato un gruppo di sicurezza che blocca il traffico in ingresso proveniente da origini diverse rispetto al cluster o ai nodi, consentendo al contempo il traffico in uscita da qualsiasi origine. Le sottoreti vengono contrassegnate in modo che Kubernetes possa implementare i load balancer.

Per creare il tuo VPC

1. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.
2. Dalla barra di navigazione, seleziona uno Regione AWS che supporti Amazon EKS.
3. Scegliere Create stack (Crea pila), With new resources (standard) (Con nuove risorse (standard)).
4. In Prerequisite - Prepare template (Prerequisito . Preparazione del modello), assicurarsi che Template is ready (Il modello è pronto) sia selezionato e quindi in Specify template (Specifica modello), selezionare Amazon S3 URL(URL Amazon S3).
5. Puoi creare un VPC che supporta solo `IPv4` o un VPC che supporta sia `IPv4` che `IPv6`. Incolla uno dei seguenti URL nell'area di testo Amazon S3 URL (URL Amazon S3) e scegli Next (Successivo):

- `IPv4`

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
amazon-eks-vpc-private-subnets.yaml
```

- `IPv4` e `IPv6`

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/
amazon-eks-ipv6-vpc-public-private-subnets.yaml
```

6. Nella pagina Specifica i dettagli dello stack immetti i relativi parametri, quindi scegli Next (Successivo).

   - Nome stack: scegli il nome per lo stack di AWS CloudFormation . Ad esempio, puoi utilizzare il nome del modello utilizzato nella fase precedente. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
   - VpcBlock: scegli una gamma `IPv4` CIDR per il tuo VPC. A ogni nodo, Pod e sistema di bilanciamento del carico che viene implementato viene assegnato un indirizzo `IPv4` da questo blocco. I valori `IPv4` di default forniscono indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificare questo comportamento. Per ulteriori informazioni, consultare VPC e dimensionamento della sottorete nella Guida per l'utente di Amazon VPC. È inoltre possibile aggiungere ulteriori blocchi CIDR al VPC una volta creato. Se stai creando un VPC `IPv6`, gli intervalli CIDR `IPv6` vengono assegnati automaticamente dallo spazio indirizzi unicast (trasmissione uno a uno) globale di Amazon.
   - PublicSubnet01Block: specifica un blocco `IPv4` CIDR per la sottorete pubblica 1. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo. Se si crea un VPC `IPv6`, questo blocco viene specificato automaticamente all'interno del modello.
   - PublicSubnet02Block: specifica un blocco `IPv4` CIDR per la sottorete pubblica 2. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo. Se si crea un VPC `IPv6`, questo blocco viene specificato automaticamente all'interno del modello.
   - PrivateSubnet01Block: specifica un blocco `IPv4` CIDR per la sottorete privata 1. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso

contrario, è comunque possibile modificarlo. Se si crea un VPC `IPv6`, questo blocco viene specificato automaticamente all'interno del modello.

- PrivateSubnet02Block: specifica un blocco `IPv4` CIDR per la sottorete privata 2. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo. Se si crea un VPC `IPv6`, questo blocco viene specificato automaticamente all'interno del modello.

7. (Facoltativo) Nella pagina Configure stack options (Configura opzioni stack), aggiungere tag alle risorse dello stack, quindi scegliere Next (Successivo).
8. Nella pagina Revisione, scegliere Crea pila.
9. Quando viene creata la pila, selezionala nella console e scegli Outputs (Uscite).
10. Registra il VpcIdper il VPC che è stato creato. Questo valore sarà necessario al momento della creazione del cluster e dei nodi.
11. Registra il SubnetIdsper le sottoreti che sono state create e se le hai create come sottoreti pubbliche o private. Saranno necessari almeno due valori al momento della creazione del cluster e dei nodi.
12. Se è stato creato un VPC `IPv4`, ignorare questo passaggio. Se è stato creato un VPC `IPv6`, è necessario abilitare l'opzione di assegnazione automatica dell'indirizzo `IPv6` per le sottoreti pubbliche create dal modello. Tale impostazione è già abilitata per le sottoreti private. Per abilitare l'impostazione, completare la procedura seguente:

    a. Apri alla console Amazon VPC all'indirizzo https://console.aws.amazon.com/vpc/.

    b. Nel pannello di navigazione a sinistra, seleziona Subnets (Sottoreti).

    c. Seleziona una delle tue sottoreti pubbliche (***stack-name*/SubnetPublic01 o *stack-name***/SubnetPublic02 contiene la parola public) e scegli Azioni, Modifica impostazioni di sottorete.

    d. Seleziona la casella di controllo Enable auto-assign **IPv6** address (Abilita l'assegnazione automatica dell'indirizzo IPv6) quindi scegli Save (Salva).

    e. Esegui nuovamente i passaggi precedenti per l'altra sottorete pubblica.

Only public subnets

Questo VPC dispone di tre sottoreti pubbliche implementate in diverse zona di disponibilità in una Regione AWS. Tutti i nodi vengono assegnati automaticamente a indirizzi `IPv4` pubblici e possono inviare e ricevere traffico Internet tramite un gateway Internet. Viene implementato un

gruppo di sicurezza che nega tutto il traffico in ingresso e consente tutto il traffico in uscita. Le sottoreti vengono contrassegnate in modo che Kubernetes possa implementare i load balancer.

Per creare il tuo VPC

1. Apri AWS CloudFormation la console all'indirizzo https://console.aws.amazon.com/cloudformation.
2. Dalla barra di navigazione, seleziona uno Regione AWS che supporti Amazon EKS.
3. Scegliere Create stack (Crea pila), With new resources (standard) (Con nuove risorse (standard)).
4. In Preparazione del modello, assicurarsi che Template is ready (Il modello è pronto) sia selezionato e quindi in Origine del modello, selezionare Amazon S3 URL.
5. Incollare il seguente URL nell'area di testo URL Amazon S3 e scegliere Successivo:

   ```
   https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-vpc-sample.yaml
   ```

6. Nella pagina Specify Details (Specifica dettagli) immetti i relativi parametri, quindi scegli Next (Successivo).

   - Nome stack: scegli il nome per lo stack di AWS CloudFormation . Ad esempio, è possibile chiamarlo ***amazon-eks-vpc-sample***. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
   - VpcBlock: scegli un blocco CIDR per il tuo VPC. A ogni nodo, Pod e sistema di bilanciamento del carico che viene implementato viene assegnato un indirizzo `IPv4` da questo blocco. I valori `IPv4` di default forniscono indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificare questo comportamento. Per ulteriori informazioni, consultare VPC e dimensionamento della sottorete nella Guida per l'utente di Amazon VPC. È inoltre possibile aggiungere ulteriori blocchi CIDR al VPC una volta creato.
   - Subnet01Block: specifica un blocco CIDR per la sottorete 1. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo.

- Subnet02Block: specifica un blocco CIDR per sottorete 2. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo.
- Subnet03Block: specifica un blocco CIDR per sottorete 3. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo.

7. (Facoltativo) Nella pagina Options (Opzioni), contrassegna con dei tag le risorse della pila. Seleziona Successivo.
8. Nella pagina Review (Revisione) scegli Create (Crea).
9. Quando viene creata la pila, selezionala nella console e scegli Outputs (Uscite).
10. Registra il VpcIdper il VPC che è stato creato. Questo valore sarà necessario al momento della creazione del cluster e dei nodi.
11. Registra il SubnetIdsper le sottoreti che sono state create. Saranno necessari almeno due valori al momento della creazione del cluster e dei nodi.
12. (Facoltativo) Qualsiasi cluster implementato su questo VPC è in grado di assegnare indirizzi `IPv4` privati a Pods e services. Se si desidera che i cluster implementati in questo VPC assegnino indirizzi `IPv6` privati a Pods e services, è necessario apportare aggiornamenti al VPC, alla sottorete, alle tabelle di instradamento e ai gruppi di sicurezza. Per ulteriori informazioni, consulta Migrazione di VPC esistenti da `IPv4` a `IPv6` nella Guida per l'utente di Amazon VPC. Amazon EKS richiede che le sottoreti abbiano l'opzione degli indirizzi `Auto-assign IPv6` abilitata. Per impostazione predefinita, tale opzione è disattivata.

Only private subnets

Questo VPC dispone di tre sottoreti private implementate in diverse zone di disponibilità nella Regione AWS. Le risorse implementate nelle sottoreti non possono accedere a Internet, né viceversa. Il modello crea endpoint VPC utilizzando AWS PrivateLink diversi endpoint a Servizi AWS cui i nodi in genere devono accedere. Per consentire ai nodi di accedere a Internet in uscita, puoi aggiungere un gateway NAT pubblico nella zona di disponibilità di ogni sottorete dopo la creazione del VPC. Viene creato un gruppo di sicurezza che blocca tutto il traffico in entrata, ad eccezione delle risorse implementate nelle sottoreti. Un gruppo di sicurezza consente inoltre tutto il traffico in uscita. Le sottoreti vengono contrassegnate in modo che Kubernetes possa implementare i load balancer interni. Se stai creando un VPC con questa configurazione, consulta Requisiti dei cluster privati per ulteriori requisiti e considerazioni.

Per creare il tuo VPC

1. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.
2. Dalla barra di navigazione, seleziona uno Regione AWS che supporti Amazon EKS.
3. Scegliere Create stack (Crea pila), With new resources (standard) (Con nuove risorse (standard)).
4. In Preparazione del modello, assicurarsi che Template is ready (Il modello è pronto) sia selezionato e quindi in Origine del modello, selezionare Amazon S3 URL.
5. Incollare il seguente URL nell'area di testo URL Amazon S3 e scegliere Successivo:

```
https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2020-10-29/amazon-eks-fully-private-vpc.yaml
```

6. Nella pagina Specify Details (Specifica dettagli), immetti i relativi parametri, quindi scegli Next (Successivo).

   - Nome stack: scegli il nome per lo stack di AWS CloudFormation . Ad esempio, è possibile chiamarlo ***`amazon-eks-fully-private-vpc`***. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
   - VpcBlock: scegli un blocco CIDR per il tuo VPC. A ogni nodo, Pod e sistema di bilanciamento del carico che viene implementato viene assegnato un indirizzo `IPv4` da questo blocco. I valori `IPv4` di default forniscono indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificare questo comportamento. Per ulteriori informazioni, consultare VPC e dimensionamento della sottorete nella Guida per l'utente di Amazon VPC. È inoltre possibile aggiungere ulteriori blocchi CIDR al VPC una volta creato.
   - PrivateSubnet01Block: specifica un blocco CIDR per la sottorete 1. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo.
   - PrivateSubnet02Block: specifica un blocco CIDR per la sottorete 2. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo.

- PrivateSubnet03Block: specifica un blocco CIDR per la sottorete 3. Il valore di default fornisce indirizzi IP sufficienti per la maggior parte delle implementazioni. In caso contrario, è comunque possibile modificarlo.

7. (Facoltativo) Nella pagina Options (Opzioni), contrassegna con dei tag le risorse della pila. Seleziona Successivo.
8. Nella pagina Review (Revisione) scegli Create (Crea).
9. Quando viene creata la pila, selezionala nella console e scegli Outputs (Uscite).
10. Registra il VpcIdper il VPC che è stato creato. Questo valore sarà necessario al momento della creazione del cluster e dei nodi.
11. Registra il SubnetIdsper le sottoreti che sono state create. Saranno necessari almeno due valori al momento della creazione del cluster e dei nodi.
12. (Facoltativo) Qualsiasi cluster implementato su questo VPC è in grado di assegnare indirizzi `IPv4` privati a Pods e services. Se si desidera che i cluster implementati in questo VPC assegnino indirizzi `IPv6` privati a Pods e services, è necessario apportare aggiornamenti al VPC, alla sottorete, alle tabelle di instradamento e ai gruppi di sicurezza. Per ulteriori informazioni, consulta Migrazione di VPC esistenti da `IPv4` a `IPv6` nella Guida per l'utente di Amazon VPC. Amazon EKS richiede che le sottoreti abbiano l'opzione degli indirizzi `Auto-assign IPv6` abilitata (disattivata per impostazione predefinita).

Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS

In questo argomento vengono descritti i requisiti relativi al gruppo di sicurezza per un cluster Amazon EKS.

Quando si crea un cluster, Amazon EKS crea un gruppo di sicurezza denominato `eks-cluster-sg-`*`my-cluster`*`-`*`uniqueID`*. Questo gruppo di sicurezza presenta le regole predefinite seguenti:

Tipo di regola	Protocollo	Porte	Origine	Destinazione
In entrata	Tutti	Tutti	Personale	
In uscita	Tutti	Tutti		0.0.0.0/0 (`IPv4`) or ::/0 (`IPv6`)

> **Important**
>
> Se il tuo cluster non necessita della regola in uscita, puoi rimuoverla. Se la rimuovi, dovrai comunque avere le regole minime elencate in Limitazione del traffico del cluster. Se rimuovi la regola in entrata, Amazon EKS la ricrea ogni volta che il cluster viene aggiornato.

Amazon EKS aggiunge i seguenti tag al gruppo di sicurezza. Se rimuovi i tag, Amazon EKS li aggiunge nuovamente al gruppo di sicurezza ogni volta che il cluster viene aggiornato.

Chiave	Valore
`kubernetes.io/cluster/ my-cluster`	`owned`
`aws:eks:cluster-name`	*my-cluster*
`Name`	`eks-cluster-sg-` *my-cluste r* `-uniqueid`

Amazon EKS crea le risorse riportate di seguito e le associa automaticamente a questo gruppo di sicurezza:

- 2-4 interfacce di rete elastiche (indicate nel resto del documento come interfacce di rete) create insieme al cluster.
- Interfacce di rete dei nodi in qualsiasi gruppo di nodi gestito creato.

Le regole predefinite consentono il flusso del traffico tra il cluster e i nodi e il traffico in uscita verso qualsiasi destinazione. Quando crei un cluster, puoi specificare i gruppi di sicurezza personali se lo desideri. In tal caso, Amazon EKS associa i gruppi di sicurezza specificati alle interfacce di rete create per il cluster, ma non li associa ai gruppi di nodi.

Puoi specificare l'ID del gruppo di sicurezza del cluster nella AWS Management Console all'interno della sezione Reti. In alternativa, puoi eseguire il comando AWS CLI seguente.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId
```

Limitazione del traffico cluster

Se è necessario limitare le porte aperte tra il cluster e i nodi, puoi rimuovere la regola in uscita predefinita e aggiungere le regole minime necessarie per il cluster. Se rimuovi la regola in ingresso predefinita, Amazon EKS la ricrea ogni volta che il cluster viene aggiornato.

Tipo di regola	Protocollo	Porta	Destinazione
In uscita	TCP	443	Gruppo di sicurezza del cluster
In uscita	TCP	10250	Gruppo di sicurezza del cluster
In uscita (DNS)	TCP e UDP	53	Gruppo di sicurezza del cluster

È inoltre necessario aggiungere regole per il traffico seguente:

- Qualsiasi protocollo e porta che si prevede di utilizzare per la comunicazione tra i nodi.
- Accesso a Internet in uscita, in modo che i nodi possano accedere alle API di Amazon EKS per l'introspezione del cluster e la registrazione dei nodi al momento del lancio. Se i nodi non hanno accesso a Internet, consulta Requisiti dei cluster privati per ulteriori considerazioni.
- Accesso ai nodi per estrarre le immagini del container da Amazon ECR o da altre API dei registri del container da cui è necessario estrarre le immagini, ad esempio DockerHub. Per ulteriori informazioni, consulta Intervalli di indirizzi IP di AWS nella Riferimenti generali di AWS.
- Accesso dei nodi ad Amazon S3.
- Sono necessarie regole separate per gli indirizzi `IPv4` e `IPv6`.

Se stai pianificando una limitazione delle regole, ti consigliamo di testare accuratamente tutti i Pods prima di applicare le regole modificate a un cluster di produzione.

Se precedentemente hai implementato un cluster con `1.14` Kubernetes e una versione della piattaforma `eks.3` o precedente, considera quanto segue:

- È probabile che siano presenti anche piani di controllo (control-plane) e gruppi di sicurezza dei nodi. Al momento della creazione, questi gruppi contenevano le regole con restrizioni elencate

nella tabella precedente, che adesso possono essere rimosse in quanto non più necessarie. Tuttavia, è necessario assicurarsi che il gruppo di sicurezza del cluster contenga le regole incluse in tali gruppi.

- Se il cluster è stato implementato utilizzando direttamente l'API o è stato creato tramite uno strumento come AWS CLI o AWS CloudFormation senza specificare alcun un gruppo di sicurezza, alle interfacce di rete del cluster create da Amazon EKS viene applicato il gruppo di sicurezza predefinito del VPC.

Componenti aggiuntivi di rete Amazon EKS

Per il cluster Amazon EKS sono disponibili diversi componenti aggiuntivi di rete.

Componenti aggiuntivi integrati

> **Note**
>
> Se crei cluster in qualsiasi modo tranne che utilizzando la console, ogni cluster viene fornito con le versioni autogestite dei componenti aggiuntivi integrati. Le versioni autogestite non possono essere gestite da AWS Management Console AWS Command Line Interface, o SDK. Sei tu a gestire la configurazione e gli aggiornamenti dei componenti aggiuntivi autogestiti.
> Consigliamo di aggiungere al cluster il componente aggiuntivo del tipo Amazon EKS anziché quello del tipo autogestito. Se crei cluster utilizzando la console , viene installata la versione Amazon EKS di questi componenti aggiuntivi.

Amazon VPC CNI plugin for Kubernetes

Questo componente aggiuntivo CNI crea interfacce di rete elastiche e le collega ai nodi Amazon EC2. Il componente aggiuntivo, inoltre, assegna un indirizzo `IPv4` o `IPv6` privato dal VPC a ogni Pod e servizio. Per impostazione predefinita, nel cluster è stato installato questo componente aggiuntivo. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.

CoreDNS

CoreDNS è un server DNS flessibile ed estensibile che può fungere da DNS del clusterKubernetes,CoreDNS. fornisce la risoluzione dei nomi per tutti i Pods del cluster. Per

impostazione predefinita, nel cluster è stato installato questo componente aggiuntivo. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo CoreDNS di Amazon EKS.

`kube-proxy`

Questo componente aggiuntivo mantiene le regole di rete sui nodi Amazon EC2 e consente la comunicazione di rete con i tuoi Pods. Per impostazione predefinita, nel cluster è stato installato questo componente aggiuntivo. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Kubernetes `kube-proxy`.

Componenti aggiuntivi di rete opzionali AWS

AWS Load Balancer Controller

Quando si distribuiscono oggetti di Kubernetes servizio di tipo diversoloadbalancer, il controller crea AWS Network Load Balancer. Quando si creano oggetti in Kubernetes ingresso, il controller crea AWS Application Load Balancer. Consigliamo di utilizzare questo controller per il provisioning di Network Load Balancer, anziché utilizzare il controller Cloud Provider legacy integrato in Kubernetes. Per ulteriori informazioni, consulta la documentazione AWS Load Balancer Controller.

AWS Controller API Gateway

Questo controller consente di connettere i servizi su più cluster Kubernetes utilizzando Kubernetes l'API del gateway. Il controller collega Kubernetes servizi in esecuzione su istanze, container e funzioni serverless Amazon EC2, utilizzando il servizio Amazon VPC Lattice. Per ulteriori informazioni, consulta documentazione Controller API Gateway AWS.

Per ulteriori informazioni sui componenti aggiuntivi, consulta Componenti aggiuntivi Amazon EKS.

Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS

Il componente aggiuntivo Amazon VPC CNI plugin for Kubernetes viene implementato su ogni nodo Amazon EC2 del cluster Amazon EKS. Il componente aggiuntivo crea interfacce di rete elastiche e le collega ai nodi Amazon EC2. Il componente aggiuntivo, inoltre, assegna un indirizzo `IPv4` o `IPv6` privato dal VPC a ogni Pod e servizio.

Una versione del componente aggiuntivo viene implementata con ogni nodo Fargate nel cluster, ma non viene aggiornata sui nodi Fargate. Altri plugin CNI compatibili sono disponibili per l'uso sui cluster Amazon EKS, ma questo è l'unico plugin CNI supportato da Amazon EKS.

La tabella seguente elenca la versione più recente del componente aggiuntivo di Amazon EKS disponibile per ogni versione Kubernetes.

Versione Kubernetes	1.30	1.29	1.28	1.27	1.26	1.25	1.24	1.23
Tipo di VPC di Amazon EKS versione CNI	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild	v1.18.2-eksbuild.1

> **Important**
>
> Se gestisci autonomamente questo componente aggiuntivo, le versioni nella tabella potrebbero non essere le stesse delle versioni autogestite disponibili. Per ulteriori informazioni sull'aggiornamento del componente aggiuntivo del tipo autogestito, consulta Aggiornamento del componente aggiuntivo autogestito.

> **Important**
>
> Per eseguire l'aggiornamento a VPC CNI v1.12.0 o versione successiva, devi prima eseguire l'aggiornamento a VPC CNI v1.7.0. È consigliabile aggiornare solo una versione secondaria alla volta.

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.

- Un provider (IAM) () esistente AWS Identity and Access Management per il tuo cluster. OpenID Connect OIDC Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- Un ruolo IAM con la policy IAM AmazonEKS_CNI_Policy (se il cluster utilizza la famiglia `IPv4`) o una policy IPv6 (se il cluster utilizza la famiglia `IPv6`) allegata. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).
- Se usi la versione `1.7.0` o successiva del Amazon VPC CNI plugin for Kubernetes e usi policy di sicurezza Pod personalizzate, consulta Eliminazione della policy di sicurezza Pod predefinita di Amazon EKSPolicy di sicurezza pod.
- **Important**

  Amazon VPC CNI plugin for Kubernetesversioni `v1.16.0` in cui è `v1.16.1` stata rimossa la compatibilità con Kubernetes le versioni `1.23` e precedenti. La versione VPC CNI `v1.16.2` ripristina la compatibilità con Kubernetes le versioni precedenti `1.23` e le specifiche CNI. `v0.4.0`

  Amazon VPC CNI plugin for Kubernetesversioni `v1.16.0` per `v1.16.1` implementare la versione delle specifiche CNI. `v1.0.0` La specifica CNI `v1.0.0` è supportata sui cluster EKS che eseguono le Kubernetes versioni o successive. `v1.24` La versione `v1.16.0` VPC CNI `v1.16.1` e le specifiche CNI `v1.0.0` non sono supportate nella versione o nelle versioni precedenti. Kubernetes `v1.23` Per ulteriori informazioni sulle `v1.0.0` specifiche CNI, vedere Container Network Interface (CNI) Specification su

Considerazioni

- Le versioni sono specificate come `major-version.minor-version.patch-version-eksbuild.build-number`.
- Verifica della compatibilità delle versioni per le varie funzionalità

  Alcune funzionalità di ciascuna versione di Amazon VPC CNI plugin for Kubernetes richiedono determinate versioni. Kubernetes Quando si utilizzano diverse funzionalità di Amazon EKS ed è necessaria una versione specifica del componente aggiuntivo, questa viene annotata nella documentazione relativa alle funzionalità. A meno che tu non abbia un motivo specifico per eseguire una versione più vecchia, consigliamo di eseguire la versione più recente.

Creare il componente aggiuntivo di Amazon EKS

Crea il componente aggiuntivo del tipo Amazon EKS.

1. Scopri qual è la versione del componente aggiuntivo attualmente installata sul cluster.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

   Di seguito viene riportato un output di esempio:

```
v1.16.4-eksbuild.2
```

2. Scopri qual è il tipo di componente aggiuntivo attualmente installato sul cluster. A seconda dello strumento con cui hai creato il cluster, al momento potresti non avere il componente aggiuntivo del tipo Amazon EKS installato sul cluster. Sostituisci *my-cluster* con il nome del cluster.

```
$ aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

   Se viene restituito il numero di versione, sul cluster è installato il componente aggiuntivo del tipo Amazon EKS e non è necessario completare i passaggi rimanenti di questa procedura. Se viene restituito un errore, sul cluster non è installato il componente aggiuntivo del tipo Amazon EKS. Completa i passaggi rimanenti di questa procedura per installarlo.

3. Salva la configurazione del componente aggiuntivo attualmente installato.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. Crea il componente aggiuntivo utilizzando la AWS CLI. Se desideri utilizzare AWS Management Console o `eksctl` creare il componente aggiuntivo, consulta Creazione di un componente aggiuntivo e specifica il nome del componente `vpc-cni` aggiuntivo. Copia il comando seguente sul tuo dispositivo. Apporta le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato.

   - Sostituisci *my-cluster* con il nome del cluster.
   - Sostituisci *v1.18.2-eksbuild.1* con la versione più recente indicata nella tabella delle versioni più recenti per la versione del cluster.

- Sostituisci *111122223333* con il tuo ID account e *AmazonEKSVPCCNIRole* con il nome del ruolo IAM esistente creato. Per specificare un ruolo, devi disporre di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno per il tuo cluster o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.18.2-eksbuild.1 \
    --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
```

Se hai applicato impostazioni personalizzate al tuo attuale componente aggiuntivo che sono in conflitto con le impostazioni predefinite del componente aggiuntivo di Amazon EKS, la creazione potrebbe non riuscire. Se la creazione non riesce, riceverai un errore che può aiutare a risolvere il problema. In alternativa, puoi aggiungere **`--resolve-conflicts OVERWRITE`** al comando precedente. Ciò consente al componente aggiuntivo di sovrascrivere le impostazioni personalizzate esistenti. Una volta creato il componente aggiuntivo, puoi aggiornarlo con le tue impostazioni personalizzate.

5. Verifica che la versione più recente del componente aggiuntivo per la versione Kubernetes del cluster sia stata aggiunta al cluster. Sostituisci *my-cluster* con il nome del cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

La creazione del componente aggiuntivo potrebbe richiedere alcuni secondi.

Di seguito viene riportato un output di esempio:

```
v1.18.2-eksbuild.1
```

6. Se hai creato impostazioni personalizzate per il componente aggiuntivo originale, prima di creare il componente aggiuntivo di Amazon EKS, utilizza la configurazione che hai salvato precedentemente per aggiornare il componente aggiuntivo di Amazon EKS con le tue impostazioni personalizzate.
7. (Facoltativo) Installa `cni-metrics-helper` sul tuo cluster. Raccoglie le informazioni sull'interfaccia di rete elastica e sull'indirizzo IP, le aggrega a livello di cluster e pubblica le metriche su Amazon. CloudWatch Per ulteriori informazioni, consulta cni-metrics-helper on. GitHub

Aggiornamento del componente aggiuntivo di Amazon EKS

Aggiorna il componente aggiuntivo del tipo Amazon EKS. Se non hai aggiunto il componente aggiuntivo del tipo Amazon EKS al tuo cluster, aggiungilo o consulta Aggiornamento del componente aggiuntivo autogestito, invece di completare questa procedura.

1. Scopri qual è la versione del componente aggiuntivo attualmente installata sul cluster. Sostituisci *my-cluster* con il nome del cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    "addon.addonVersion" --output text
   ```

   Di seguito viene riportato un output di esempio:

   ```
   v1.16.4-eksbuild.2
   ```

   Se la versione restituita è la stessa della versione Kubernetes del cluster nella tabella delle versioni più recenti, la versione più recente è già installata nel cluster e non è necessario completare il resto di questa procedura. Se invece di un numero di versione ricevi un errore, il componente aggiuntivo del tipo Amazon EKS non è installato sul cluster. È necessario creare il componente aggiuntivo prima di poterlo aggiornare con questa procedura.

2. Salva la configurazione del componente aggiuntivo attualmente installato.

   ```
   kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
   ```

3. Aggiorna il componente aggiuntivo utilizzando la AWS CLI. Se desideri utilizzare AWS Management Console o `eksctl` aggiornare il componente aggiuntivo, consulta. Aggiornamento di un componente aggiuntivo Copia il comando seguente sul tuo dispositivo. Apporta le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato.

   - Sostituisci *my-cluster* con il nome del cluster.
   - Sostituisci *v1.18.2-eksbuild.1* con la versione più recente indicata nella tabella delle versioni più recenti per la versione del cluster.
   - Sostituisci *111122223333* con il tuo ID account e *AmazonEKSVPCCNIRole* con il nome del ruolo IAM esistente creato. Per specificare un ruolo, devi disporre di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno per il tuo cluster o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.

- L'opzione **`--resolve-conflicts`** *`CONSERVA`* mantiene i valori di configurazione esistenti per il componente aggiuntivo. Se hai impostato valori personalizzati per le impostazioni del componente aggiuntivo e non utilizzi questa opzione, Amazon EKS sovrascrive i tuoi valori con quelli predefiniti. Se utilizzi questa opzione, è preferibile testare eventuali modifiche ai campi e ai valori su un cluster non di produzione prima di aggiornare il componente aggiuntivo sul cluster di produzione. Se modifichi questo valore in `OVERWRITE`, tutte le impostazioni vengono modificate nei valori predefiniti di Amazon EKS. Se hai impostato valori personalizzati per un'impostazione qualunque, è possibile che vengano sovrascritti con i valori predefiniti di Amazon EKS. Se modifichi questo valore in `none`, Amazon EKS non modifica il valore di alcuna impostazione, ma l'aggiornamento potrebbe non riuscire. Se l'aggiornamento non riesce, riceverai un messaggio di errore che ti aiuterà a risolvere il conflitto.
- Se non stai aggiornando un'impostazione di configurazione, rimuovi **`--configuration-values '{`***`"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}`***`}'`** dal comando. Se stai aggiornando un'impostazione di configurazione, sostituisci *`"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}`* con l'impostazione che desideri impostare. In questo esempio, la variabile di ambiente `AWS_VPC_K8S_CNI_EXTERNALSNAT` è impostata su `true`. Il valore specificato deve essere valido per lo schema di configurazione. Se non conosci lo schema di configurazione**`aws eks describe-addon-configuration --addon-name vpc-cni --addon-version`** ***`v1.18.2-eksbuild.1`***, esegui sostituendo *`v1.18.2-eksbuild.1`* con il numero di versione del componente aggiuntivo di cui vuoi vedere la configurazione. Lo schema viene restituito nell'output. Se disponi di una configurazione personalizzata, desideri rimuoverla e reimpostare i valori di tutte le impostazioni ai valori predefiniti di Amazon EKS, rimuovi *`"env":{"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}`* dal comando, in modo da avere un **`{}`** vuoto. Per una spiegazione di ciascuna impostazione, consulta Variabili di configurazione CNI su. GitHub

  ```
  aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
  version v1.18.2-eksbuild.1 \
      --service-account-role-arn arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
   \
      --resolve-conflicts PRESERVE --configuration-values '{"env":
  {"AWS_VPC_K8S_CNI_EXTERNALSNAT":"true"}}'
  ```

  Il completamento dell'aggiornamento potrebbe richiedere alcuni secondi.

4. Conferma che la versione del componente aggiuntivo sia stata aggiornata. Sostituisci *`my-cluster`* con il nome del cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

Il completamento dell'aggiornamento potrebbe richiedere alcuni secondi.

Di seguito viene riportato un output di esempio:

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.18.2-eksbuild.1",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/vpc-
cni/74c33d2f-b4dc-8718-56e7-9fdfa65d14a9",
        "createdAt": "2023-04-12T18:25:19.319000+00:00",
        "modifiedAt": "2023-04-12T18:40:28.683000+00:00",
        "serviceAccountRoleArn":
 "arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole",
        "tags": {},
        "configurationValues": "{\"env\":{\"AWS_VPC_K8S_CNI_EXTERNALSNAT\":\"true
\"}}"
    }
}
```

Aggiornamento del componente aggiuntivo autogestito

> **Important**
>
> Consigliamo di aggiungere al cluster il componente aggiuntivo del tipo Amazon EKS anziché quello del tipo autogestito. Se non conosci bene le differenze tra i due tipi, consulta la pagina the section called “Componenti aggiuntivi di Amazon EKS”. Per ulteriori informazioni sull'aggiunta di un componente aggiuntivo di Amazon EKS al cluster, consulta the section called “Creazione di un componente aggiuntivo”. Se non riesci a utilizzare il componente aggiuntivo Amazon EKS, ti consigliamo di segnalare un problema sul motivo per cui non puoi farlo all'archivio della roadmap GitHub di Containers.

1. Verifica che sul cluster non sia installato il tipo Amazon EKS del componente aggiuntivo. Sostituisci *my-cluster* con il nome del tuo cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

   Se viene restituito un messaggio di errore, sul cluster non è installato il componente aggiuntivo del tipo Amazon EKS. Per gestire automaticamente il componente aggiuntivo, completa i passaggi rimanenti di questa procedura per aggiornare il componente aggiuntivo. Se viene restituito il numero di versione, sul cluster è installato il tipo Amazon EKS del componente aggiuntivo. Per aggiornarlo, utilizza la procedura descritta in Aggiornamento di un componente aggiuntivo anziché questa. Se non conosci bene le differenze tra i due tipi di componente aggiuntivo, consulta Componenti aggiuntivi Amazon EKS.

2. Scopri qual è la versione dell'immagine di container attualmente installata sul cluster.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

   Di seguito viene riportato un output di esempio:

```
v1.16.4-eksbuild.2
```

   L'output potrebbe non includere il numero di build.

3. Effettua il backup delle impostazioni correnti in modo da poter riprodurre la medesima configurazione dopo avere aggiornato la versione.

```
kubectl get daemonset aws-node -n kube-system -o yaml > aws-k8s-cni-old.yaml
```

4. Per rivedere le versioni disponibili e familiarizzare con le modifiche apportate alla versione verso cui vuoi eseguire l'aggiornamento, consulta la pagina `releases` su GitHub. Tieni presente che ti consigliamo di eseguire l'aggiornamento allo stesso. `major` `minor`. `patch`versione elencata nella tabella delle ultime versioni disponibili, anche se le versioni successive sono disponibili su GitHub.. Le versioni di build elencate nella tabella non sono specificate nelle versioni autogestite elencate in GitHub. Aggiorna la versione completando le attività in una delle seguenti opzioni:

- Se non disponi di impostazioni personalizzate per il componente aggiuntivo, esegui il comando sotto l'`To apply this release:`intestazione GitHub relativa alla versione a cui stai eseguendo l'aggiornamento.
- Se disponi di impostazioni personalizzate, scarica il file manifesto con il comando seguente. Modifica *`https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.18.2/config/master/aws-k8s-cni.yaml`* con l'URL della versione a GitHub cui stai effettuando l'aggiornamento.

  ```
  curl -O https://raw.githubusercontent.com/aws/amazon-vpc-cni-k8s/v1.18.2/config/master/aws-k8s-cni.yaml
  ```

  Se necessario, modifica il manifesto con le impostazioni personalizzate del backup che hai effettuato in una fase precedente, quindi applica il file modificato al tuo cluster. Se i tuoi nodi non hanno accesso ai repository privati Amazon EKS Amazon ECR da cui vengono estratte le immagini (vedi le righe che iniziano con `image:` nel manifesto), dovrai scaricare le immagini, copiarle nel tuo repository e modificare il manifesto per estrarre le immagini dal repository. Per ulteriori informazioni, consulta Copia di un'immagine di container da un repository a un altro.

  ```
  kubectl apply -f aws-k8s-cni.yaml
  ```

5. Verifica che la nuova versione sia ora installata nel cluster.

   ```
   kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
    | cut -d : -f 3
   ```

   Di seguito viene riportato un output di esempio:

   ```
   v1.18.2
   ```

6. (Facoltativo) Installa `cni-metrics-helper` sul tuo cluster. Raccoglie le informazioni sull'interfaccia di rete elastica e sull'indirizzo IP, le aggrega a livello di cluster e pubblica le metriche su Amazon. CloudWatch Per ulteriori informazioni, consulta cni-metrics-helper on. GitHub

Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA)

Il Amazon VPC CNI plugin for Kubernetes è il plug-in per reti Pod nei cluster Amazon EKS. Il plug-in è responsabile dell'allocazione degli indirizzi IP VPC sui nodi Kubernetes e della configurazione delle reti necessarie per Pods in ogni nodo. Il plug-in:

- Richiede autorizzazioni AWS Identity and Access Management (IAM). Se il cluster utilizza la famiglia IPv4, le autorizzazioni sono specificate nella politica gestita. `AmazonEKS_CNI_Policy` AWS Se il cluster utilizza la famiglia IPv6, le autorizzazioni devono essere aggiunte a una policy IAM creata dall'utente.È possibile collegare questa policy al ruolo del nodo IAM Amazon EKS o a un ruolo IAM separato. Si consiglia di assegnarlo a un ruolo separato, come descritto in questo argomento.
- Crea ed è configurato per l'utilizzo di un account di servizio Kubernetes denominato `aws-node` quando viene implementato. L'account di servizio è associato a un `clusterrole` Kubernetes denominato `aws-node`, a cui vengono assegnate le autorizzazioni Kubernetes necessarie.

> **Note**
>
> I Pods per il Amazon VPC CNI plugin for Kubernetes hanno accesso alle autorizzazioni assegnate al ruolo IAM del nodo Amazon EKS, a meno che non si blocchi l'accesso al servizio di metadati a IMDS. Per ulteriori informazioni, consulta Limita l'accesso al profilo dell'istanza assegnato al nodo (worker).

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.
- Un provider AWS Identity and Access Management (IAM) OpenID Connect (OIDC) esistente per il tuo cluster. Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.

Fase 1: creazione di un ruolo IAM per Amazon VPC CNI plugin for Kubernetes

Per creare il ruolo IAM

1. Determina la famiglia IP del cluster.

   ```
   aws eks describe-cluster --name my-cluster | grep ipFamily
   ```

   Di seguito viene riportato un output di esempio:

   ```
   "ipFamily": "ipv4"
   ```

   L'output potrebbe invece restituire `ipv6`.

2. Crea il ruolo IAM. Puoi usare `eksctl` or `kubectl` e the AWS CLI per creare il tuo ruolo IAM.

   eksctl

   Crea un ruolo IAM e collega ad esso la policy IAM con il comando corrispondente alla famiglia IP del cluster. Il comando crea e distribuisce uno AWS CloudFormation stack che crea un ruolo IAM, allega la policy specificata e annota l'account di `aws-node` Kubernetes servizio esistente con l'ARN del ruolo IAM creato.

   - `IPv4`

     Sostituire *my-cluster* con il proprio valore.

     ```
     eksctl create iamserviceaccount \
         --name aws-node \
         --namespace kube-system \
         --cluster my-cluster \
         --role-name AmazonEKSVPCCNIRole \
         --attach-policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
         --override-existing-serviceaccounts \
         --approve
     ```

   - `IPv6`

     Sostituire *my-cluster* con il proprio valore. Sostituisci *111122223333* con il tuo ID account e *AmazonEKS_CNI_IPv6_Policy* con il nome della policy `IPv6`. Se non disponi di una policy `IPv6`, consulta Creare una policy IAM per i cluster che utilizzano la famiglia

IPv6 per crearne una. Per utilizzare IPv6 con il cluster, questo deve soddisfare diversi requisiti. Per ulteriori informazioni, consulta IPv6indirizzi per cluster Pods e services.

```
eksctl create iamserviceaccount \
    --name aws-node \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKSVPCCNIRole \
    --attach-policy-arn
 arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
    --override-existing-serviceaccounts \
    --approve
```

kubectl and the AWS CLI

1. Visualizza l'URL del provider OIDC del cluster.

   ```
   aws eks describe-cluster --name my-cluster --query
    "cluster.identity.oidc.issuer" --output text
   ```

   Di seguito viene riportato un output di esempio:

   ```
   https://oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE
   ```

   Se non viene restituito alcun output, devi creare un provider OIDC IAM per il cluster.

2. Copia i contenuti seguenti in un file denominato *vpc-cni-trust-policy.json*. Sostituisci *111122223333* con il tuo ID account e *EXAMPLED539D4633E53DE1B71EXAMPLE* con l'output restituito nel passaggio precedente. Sostituiscilo *region-code* con quello in cui si trova il Regione AWS cluster.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Federated": "arn:aws:iam::111122223333:oidc-provider/
   oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
               },
   ```

```
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:aws-node"
                }
            }
        }
    ]
}
```

3. Crea il ruolo. Puoi sostituire *AmazonEKSVPCCNIRole* con un nome a tua scelta.

```
aws iam create-role \
  --role-name AmazonEKSVPCCNIRole \
  --assume-role-policy-document file://"vpc-cni-trust-policy.json"
```

4. Allega la policy IAM richiesta al ruolo. Esegui il comando corrispondente alla famiglia IP del cluster.

   - IPv4

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
       --role-name AmazonEKSVPCCNIRole
     ```

   - IPv6

     Sostituisci *111122223333* con il tuo ID account e *AmazonEKS_CNI_IPv6_Policy* con il nome della policy IPv6. Se non disponi di una policy IPv6, consulta Creare una policy IAM per i cluster che utilizzano la famiglia IPv6 per crearne una. Per utilizzare IPv6 con il cluster, questo deve soddisfare diversi requisiti. Per ulteriori informazioni, consulta IPv6indirizzi per cluster Pods e services.

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
       --role-name AmazonEKSVPCCNIRole
     ```

5. Esegui il comando seguente per annotare l'account di servizio `aws-node` con l'ARN del ruolo IAM creato in precedenza. Sostituisci i *`example values`* con i valori in tuo possesso.

```
kubectl annotate serviceaccount \
    -n kube-system aws-node \
    eks.amazonaws.com/role-
arn=arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
```

3. (Facoltativo) Configura il tipo di AWS Security Token Service endpoint utilizzato dal tuo account Kubernetes di servizio. Per ulteriori informazioni, consulta Configurare l' AWS Security Token Service endpoint per un account di servizio.

Fase 2: re-implementazione dei Pods per il Amazon VPC CNI plugin for Kubernetes

1. Elimina e ricrea i Pods esistenti associati all'account di servizio per applicare le variabili di ambiente delle credenziali. L'annotazione non viene applicata ai Pods attualmente in esecuzione senza l'annotazione. Il comando seguente elimina i Pods `aws-node` DaemonSet esistenti e li implementa con l'annotazione dell'account di servizio.

```
kubectl delete Pods -n kube-system -l k8s-app=aws-node
```

2. Conferma che tutti i Pods sono stati riavviati.

```
kubectl get pods -n kube-system -l k8s-app=aws-node
```

3. Descrivi uno dei Pods e verifica che le variabili di ambiente `AWS_WEB_IDENTITY_TOKEN_FILE` e `AWS_ROLE_ARN`esistano. Sostituisci *`cpjw7`* con il nome di uno dei Pods restituiti nell'output del passaggio precedente.

```
kubectl describe pod -n kube-system aws-node-cpjw7 | grep 'AWS_ROLE_ARN:\|
AWS_WEB_IDENTITY_TOKEN_FILE:'
```

Di seguito viene riportato un output di esempio:

```
AWS_ROLE_ARN:                 arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:  /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

```
      AWS_ROLE_ARN:
 arn:aws:iam::111122223333:role/AmazonEKSVPCCNIRole
      AWS_WEB_IDENTITY_TOKEN_FILE:            /var/run/secrets/eks.amazonaws.com/
serviceaccount/token
```

Vengono restituiti due set di risultati duplicati perché il Pod contiene due container. I due container hanno gli stessi valori.

Se Pod si utilizza l'endpoint Regione AWS al, nell'output precedente viene restituita anche la riga seguente.

```
AWS_STS_REGIONAL_ENDPOINTS=regional
```

Fase 3: rimozione della policy CNI dal ruolo IAM del nodo

Se al tuo ruolo IAM del nodo Amazon EKS è attualmente associata la IPv6policy AmazonEKS_CNI_Policy IAM (IPv4) o una policy e hai creato un ruolo IAM separato, invece hai collegato la policy ad esso e l'hai assegnata all'account di aws-node Kubernetes servizio, ti consigliamo di rimuovere la policy dal ruolo del nodo con il AWS CLI comando che corrisponde alla famiglia IP del tuo cluster. Sostituisci *AmazonEKSNodeRole* con il nome del ruolo del nodo.

- IPv4

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
```

- IPv6

  Sostituisci *111122223333* con il tuo ID account e *AmazonEKS_CNI_IPv6_Policy* con il nome della policy IPv6.

```
aws iam detach-role-policy --role-name AmazonEKSNodeRole --policy-arn
 arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy
```

Creare una policy IAM per i cluster che utilizzano la famiglia **IPv6**

Se hai creato un cluster che utilizza la famiglia IPv6 e sul cluster è configurata la versione 1.10.1 o successiva del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes, puoi creare una

policy IAM che puoi assegnare a un ruolo IAM. Se disponi di un cluster che non hai configurato con la famiglia IPv6 al momento della creazione, prima di utilizzare IPv6, sarà necessario creare un nuovo cluster. Per ulteriori informazioni sull'uso di IPv6 con il cluster, consulta IPv6indirizzi per cluster Pods e services.

1. Copia il testo seguente e salvalo in un file denominato *vpc-cni-ipv6-policy*.json.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

2. Creare la policy IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-document
 file://vpc-cni-ipv6-policy.json
```

Scelta dei casi d'uso della rete di Pod

Amazon VPC CNI plugin for Kubernetes fornisce rete per i Pods. La tabella seguente consente di comprendere quali casi d'uso di rete puoi utilizzare in maniera combinata e le impostazioni di Amazon VPC CNI plugin for Kubernetes che puoi utilizzare con diversi tipi di nodi Amazon EKS. Tutte le informazioni contenute nella tabella si applicano solo ai nodi `IPv4` Linux.

Tipo di nodo di Amazon EKS	Amazon EC2			Fargate
Caso d'uso	Indirizzi IP individuali assegnati all'interfaccia di rete	Prefissi IP assegnati all'interfaccia di rete	Gruppi di sicurezza per Pods	
Rete personalizzata per i pod – Assegna indirizzi IP da una sottorete diversa dalla sottorete del nodo	Sì	Sì	Sì	Sì (subnet controllate tramite profilo Fargate)
SNAT per Pods	Sì (di default è `false`)	Sì (di default è `false`)	Sì (`true` solo)	Sì (`true` solo)
Funzionalità				
Ambito di un gruppo di sicurezza	Nodo	Nodo	Pod (se hai impostato `POD_SECURITY_GROUP_ENFORCING_MODE =standard` e `AWS_VPC_K8S_CNI_EXTERNALSNA`	Pod

Tipo di nodo di Amazon EKS	Amazon EC2			Fargate
Caso d'uso	Indirizzi IP individuali assegnati all'interfaccia di rete	Prefissi IP assegnati all'interfaccia di rete	Gruppi di sicurezza per Pods	
			`T =false`, il traffico destinato agli endpoint esterni al VPC utilizza i gruppi di sicurezza del nodo, non i gruppi di sicurezza del Pod's)	
Tipi di sottoreti Amazon VPC	IP pubblici e privati	IP pubblici e privati	Solo privati	Solo privati
Policy di rete (CNI di VPC)	Compatible	Compatible	Compatible Solo con versione `1.14.0` o successiva del plug-in CNI di Amazon VPC	Non supportato
Densità pod per nodo	Media	Elevata	Bassa	One
Ora di avvio Pod	Migliorata	Migliore	Buona	Moderata
Impostazioni del plug-in Amazon VPC CNI (per ulteriori informazioni su ciascuna impostazione, consulta amazon-vpc-cni-k 8s on) GitHub				

Tipo di nodo di Amazon EKS	Amazon EC2			Fargate
Caso d'uso	Indirizzi IP individuali assegnati all'interfaccia di rete	Prefissi IP assegnati all'interfaccia di rete	Gruppi di sicurezza per Pods	
`WARM_ENI_TARGET`	Sì	Non applicabile	Non applicabile	Non applicabile
`WARM_IP_TARGET`	Sì	Sì	Non applicabile	Non applicabile
`MINIMUM_IP_TARGET`	Sì	Sì	Non applicabile	Non applicabile
`WARM_PREFIX_TARGET`	Non applicabile	Sì	Non applicabile	Non applicabile

Note

- Non è possibile utilizzare `IPv6` con una rete personalizzata.
- Gli indirizzi `IPv6` non sono tradotti, quindi SNAT non viene applicato.
- Il flusso di traffico da e verso i Pods con gruppi di sicurezza associati non è soggetto all'applicazione della policy di rete Calico ed è limitato esclusivamente all'applicazione dei gruppi di sicurezza Amazon VPC.
- Se utilizzi l'applicazione delle policy di Calico rete, ti consigliamo di impostare la variabile di ambiente in `ANNOTATE_POD_IP` modo da `true` evitare un problema noto con. Kubernetes Per utilizzare questa funzionalità, è necessario aggiungere `patch` l'autorizzazione per i pod a. `aws-node` ClusterRole Tieni presente che l'aggiunta di autorizzazioni per le patch `aws-node` DaemonSet aumenta l'ambito di sicurezza del plug-in. Per ulteriori informazioni, consulta ANNOTATE_POD_IP nel repository VPC CNI attivo. GitHub
- I prefissi IP e gli indirizzi IP sono associati alle interfacce di rete elastiche Amazon EC2 standard. Ai pod che richiedono gruppi di sicurezza specifici viene assegnato l'indirizzo IP primario di un'interfaccia di rete di filiali. Puoi combinare i Pods ottenendo gli indirizzi IP o

gli indirizzi IP dai prefissi IP con i Pods che ottengono interfacce di rete di filiali sullo stesso nodo.

Nodi Windows

Ogni nodo supporta una sola interfaccia di rete. Puoi utilizzare indirizzi `IPv4` e prefissi `IPv4` secondari. Per impostazione predefinita, il numero di indirizzi `IPv4` disponibili sul nodo è uguale al numero di indirizzi `IPv4` secondari meno uno che puoi assegnare a ogni interfaccia di rete elastica. Tuttavia, puoi aumentare gli indirizzi `IPv4` disponibili e la densità dei Pod sul nodo abilitando i prefissi IP. Per ulteriori informazioni, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2.

Le policy di rete Calico sono supportate da Windows. Non puoi utilizzare gruppi di sicurezza per Pods o reti personalizzate su Windows.

`IPv6`indirizzi per cluster Pods e services

Per impostazione predefinita, Kubernetes assegna indirizzi `IPv4` ai Pods e ai services. Invece di assegnare indirizzi `IPv4` a Pods e services, è possibile configurare il cluster in modo da assegnare loro indirizzi `IPv6`. Amazon EKS non supporta Pods o services dual-stack, anche se Kubernetes non ha la versione `1.23` e successive. Di conseguenza, non è possibile assegnare sia indirizzi `IPv4` che `IPv6` a Pods e services.

È possibile selezionare la famiglia IP che si desidera utilizzare per il cluster al momento della creazione. Non sarà possibile modificare il nome dopo aver creato il cluster.

Considerazioni sull'utilizzo della **`IPv6`** famiglia per il cluster

- È necessario creare un nuovo cluster e specificare che vuoi utilizzare la famiglia `IPv6` per quel cluster. Non è possibile abilitare la famiglia `IPv6` per un cluster aggiornato da una versione precedente. Per istruzioni su come creare un nuovo cluster, consultare Creazione di un cluster Amazon EKS.
- La versione del componente aggiuntivo CNI di Amazon VPC implementata sul cluster deve essere la versione `1.10.1` o successiva. Per impostazione predefinita, viene distribuita questa versione o una versione successiva. Dopo aver implementato il componente aggiuntivo, non sarà possibile eseguire il downgrade del componente aggiuntivo CNI di Amazon VPC a una versione inferiore a `1.10.1` senza rimuovere tutti i nodi in tutti i gruppi di nodi nel cluster.

- I Pods e i services Windows non sono supportati.
- Se si utilizzano i nodi Amazon EC2, è necessario configurare il componente aggiuntivo CNI di Amazon VPC con delega del prefisso IP e `IPv6`. Se durante la creazione del cluster si sceglie la famiglia `IPv6`, la versione `1.10.1` del componente aggiuntivo utilizza per impostazione predefinita questa configurazione. Questo è il caso di un componente aggiuntivo autogestito o Amazon EKS. Per ulteriori informazioni sulla delega dei prefissi IP, consultare Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2.
- Quando si crea un cluster, il VPC e le sottoreti specificate devono avere un blocco CIDR `IPv6` assegnato al VPC e alle sottoreti specificate. Devono inoltre disporre di un blocco CIDR `IPv4` assegnato. Questo perché, anche se si desidera utilizzare solo `IPv6`, il funzionamento di un VPC richiede comunque un blocco CIDR `IPv4`. Per ulteriori informazioni, consulta Associazione di un blocco CIDR `IPv6` al VPC nella Guida per l'utente di Amazon VPC.
- Quando si creano cluster e nodi, è necessario specificare le sottoreti configurate per l'assegnazione automatica degli indirizzi `IPv6`. In caso contrario, non sarà possibile implementare cluster e nodi. Per impostazione predefinita, questa configurazione è disabilitata. Per ulteriori informazioni, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv6` per la sottorete nella Guida per l'utente di Amazon VPC.
- Le tabelle di instradamento assegnate alle sottoreti devono avere percorsi per gli indirizzi `IPv6`. Per ulteriori informazioni, consulta Migrare a `IPv6` nella Guida per l'utente di Amazon VPC.
- I gruppi di sicurezza devono consentire indirizzi `IPv6`. Per ulteriori informazioni, consulta Migrare a `IPv6` nella Guida per l'utente di Amazon VPC.
- Puoi utilizzarlo solo `IPv6` con nodi Amazon EC2 o Fargate AWS basati su Nitro.
- Non è possibile utilizzare `IPv6` con Gruppi di sicurezza per Pods con nodi Amazon EC2. Tuttavia, è possibile utilizzarlo con i nodi Fargate. Se sono necessari gruppi di sicurezza separati per singoli Pods, continua a utilizzare la famiglia `IPv4` con i nodi Amazon EC2 o utilizza invece i nodi Fargate.
- Per ridurre l'esaurimento dell'indirizzo IP, puoi utilizzare l'indirizzo `IPv6` anche se in precedenza hai impiegato una rete personalizzata. Non è possibile utilizzare una rete personalizzata con `IPv6`. Se si utilizza una rete personalizzata per l'isolamento della rete, potrebbe essere necessario continuare a utilizzare la rete personalizzata e la famiglia `IPv4` per i cluster.
- Non è possibile utilizzare `IPv6` con AWS Outposts.
- A Pods e services viene assegnato solo un indirizzo `IPv6`, e non un indirizzo `IPv4`. Siccome i Pods sono in grado di comunicare con gli endpoint `IPv4` tramite NAT sull'istanza stessa, DNS64 e NAT64 non sono necessari. Se il traffico richiede un indirizzo IP pubblico, il traffico è quindi l'indirizzo di rete di origine tradotto in un IP pubblico.

- L'indirizzo IPv6 di origine di un Pod non è l'indirizzo di rete di origine tradotto nell'indirizzo IPv6 del nodo quando si comunica all'esterno del VPC. Viene instradato utilizzando un gateway Internet o gateway Internet egress-only.
- A tutti i nodi viene assegnato un indirizzo IPv4 e uno IPv6.
- Driver CSI per Amazon FSx for Lustre non è supportato.
- È possibile utilizzare la versione 2.3.1 o successiva del AWS Load Balancer Controller per bilanciare il carico dell'applicazione o del traffico di rete IPv6 Pods in modalità IP, ma non in modalità istanza. Per ulteriori informazioni, consulta Che cosa è la AWS Load Balancer Controller?.
- È necessario allegare una policy IAM IPv6 al ruolo IAM del nodo o del CNI. Tra i due, consigliamo di allegarlo a un ruolo IAM CNI. Per ulteriori informazioni, consulta Creare una policy IAM per i cluster che utilizzano la famiglia IPv6 e Fase 1: creazione di un ruolo IAM per Amazon VPC CNI plugin for Kubernetes.
- Ogni Pod Fargate riceve un indirizzo IPv6 dal CIDR specificato per la sottorete in cui è stato implementato. L'unità hardware sottostante che esegue i Pods Fargate ottiene un indirizzo IPv4 e uno IPv6 univoci dai CIDR assegnati alla sottorete in cui è implementata l'unità hardware.
- Ti consigliamo di eseguire una valutazione approfondita delle tue applicazioni, dei componenti aggiuntivi di Amazon EKS e AWS dei servizi con cui ti integri prima di distribuire IPv6 i cluster. Questo per garantire che tutto funzioni come previsto con IPv6.
- L'utilizzo dell'endpoint IPv6 di Amazon EC2 Instance Metadata Service non è supportato con Amazon EKS.
- Quando crei un gruppo di nodi autogestito in un cluster che utilizza la famiglia IPv6, i dati utente devono includere i seguenti BootstrapArguments per il file bootstrap.sh che viene eseguito all'avvio del nodo. Sostituisci *your-cidr* con l'intervallo CIDR IPv6 del VPC del cluster.

```
--ip-family ipv6 --service-ipv6-cidr your-cidr
```

  Se non conosci l'IPv6CIDRintervallo per il tuo cluster, puoi visualizzarlo con il seguente comando (richiede la AWS CLI versione 2.4.9 o successiva).

```
aws eks describe-cluster --name my-cluster --query
 cluster.kubernetesNetworkConfig.serviceIpv6Cidr --output text
```

Implementazione di un cluster **IPv6** e nodi Amazon Linux gestiti

In questo tutorial, implementi un Amazon VPC IPv6, un cluster Amazon EKS con la famiglia IPv6 e un gruppo di nodi gestiti con nodi Amazon Linux per Amazon EC2. Non puoi implementare i nodi Windows per Amazon EC2 in un cluster IPv6. È inoltre possibile implementare i nodi Fargate nel cluster, sebbene tali istruzioni non siano fornite in questo argomento per semplicità.

Prima di creare un cluster da utilizzare in produzione, è consigliabile acquisire familiarità con tutte le impostazioni e implementare un cluster con le impostazioni che soddisfano i tuoi requisiti. Per ulteriori informazioni, consultare Creazione di un cluster Amazon EKS, Gruppi di nodi gestiti e le considerazioni per questo argomento. Alcune impostazioni possono essere abilitate solo quando viene creato il cluster.

Prerequisiti

Prima di iniziare questo tutorial, è necessario installare e configurare i seguenti strumenti e risorse necessarie per creare e gestire un cluster Amazon EKS.

- Lo strumento a riga di comando kubectl è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è 1.29, puoi usare kubectl versione 1.28, 1.29 o 1.30. Per installare o aggiornare kubectl, consulta Installazione o aggiornamento di kubectl:
- Il responsabile della sicurezza IAM che stai utilizzando deve disporre delle autorizzazioni per lavorare con i ruoli IAM di Amazon EKS, i ruoli collegati ai servizi AWS CloudFormation, un VPC e le risorse correlate. Per ulteriori informazioni, consulta Operazioni, risorse e chiavi di condizione per Amazon Elastic Kubernetes Service e Utilizzo di ruoli collegati ai servizi nella Guida per l'utente di IAM.

Vengono fornite le procedure per creare le risorse con eksctl o la AWS CLI. Puoi anche distribuire le risorse utilizzando AWS Management Console, ma per semplicità tali istruzioni non sono fornite in questo argomento.

eksctl

Prerequisito

eksctl versione 0.183.0 o successiva installata sul computer. Per installarlo o aggiornarlo, consulta la sezione Installation nella documentazione di eksctl.

Per implementare un cluster **IPv6** con **eksctl**

1. Crea il file `ipv6-cluster.yaml`. Copia il comando seguente sul tuo dispositivo. Apportare le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato:

   - Sostituisci *my-cluster* con un nome da assegnare al cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
   - Sostituisci *region-code* con qualsiasi Regione AWS supportata da Amazon EKS. Per un elenco di Regioni AWS, consulta gli endpoint e le quote di Amazon EKS nella guida di riferimento AWS generale.
   - Il valore per `version` con la versione del tuo cluster. Per ulteriori informazioni, consulta la Versione Kubernetes Amazon EKS supportata.
   - Sostituisci *my-nodegroup* con un nome per il gruppo di nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura.
   - Sostituisci *t3.medium* con qualsiasi tipo di istanza di AWS Nitro System.

```
cat >ipv6-cluster.yaml <<EOF
---
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "X.XX"

kubernetesNetworkConfig:
  ipFamily: IPv6

addons:
  - name: vpc-cni
    version: latest
  - name: coredns
    version: latest
  - name: kube-proxy
    version: latest
```

```
iam:
  withOIDC: true

managedNodeGroups:
  - name: my-nodegroup
    instanceType: t3.medium
EOF
```

2. Creare il cluster.

```
eksctl create cluster -f ipv6-cluster.yaml
```

La creazione di cluster richiede diversi minuti. Non procedere finché non viene visualizzata l'ultima riga di output, simile alla seguente.

```
[...]
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

3. Confermare che ai Pods di default siano assegnati indirizzi IPv6.

```
kubectl get pods -n kube-system -o wide
```

Di seguito viene riportato un output di esempio:

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                             NODE
 NOMINATED NODE   READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
aws-node-t74jh             1/1     Running   0          5m32s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

```
kube-proxy-btpbk           1/1     Running   0          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

4. Confermare che ai servizi di default siano assegnati indirizzi IPv6.

   ```
   kubectl get services -n kube-system -o wide
   ```

   Di seguito viene riportato un output di esempio:

   ```
   NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
    SELECTOR
   kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m
    k8s-app=kube-dns
   ```

5. (Facoltativo) Implementa un'applicazione di esempio o implementa il AWS Load Balancer Controller e un'applicazione di esempio per eseguire il bilanciamento del carico dell'applicazione o del traffico di rete verso i IPv6 Pods.
6. Una volta terminato con il cluster e i nodi creati per questo tutorial, eliminare le risorse create con il comando riportato di seguito.

   ```
   eksctl delete cluster my-cluster
   ```

AWS CLI

Prerequisito

Versione 2.12.3 o successiva o versione 1.27.160 o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul tuo dispositivo o. AWS CloudShell Per verificare la versione attuale, usa **aws --version | cut -d / -f2 | cut -d ' ' -f1**. I programmi di gestione dei pacchetti, come yum, apt-get o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per

l'AWS CloudShell utente. Se si utilizza la AWS CloudShell, potrebbe essere necessario installare la versione `2.12.3` o una versione successiva `1.27.160` o successiva di AWS CLI, poiché la AWS CLI versione predefinita installata in AWS CloudShell potrebbe essere una versione precedente.

> **Important**
>
> - È necessario che tutti i passaggi di questa guida siano completati dallo stesso utente. Esegui il comando seguente per controllare l'utente corrente:
>
>   ```
>   aws sts get-caller-identity
>   ```
>
> - È necessario completare tutti i passaggi di questa procedura nella stessa shell. Diversi passaggi utilizzano le variabili impostate nelle fasi precedenti. Le fasi che utilizzano le variabili non funzioneranno correttamente se i valori delle variabili sono impostati in una shell diversa. Se si utilizza la AWS CloudShell per completare la procedura seguente, tenere presente che se non si interagisce con la tastiera o il puntatore per circa 20-30 minuti, la sessione di shell termina. I processi in esecuzione non sono considerati interazioni.
> - Le istruzioni sono scritte per la shell Bash e per altre shell potrebbero essere necessarie ulteriori regolazioni.

Per creare il cluster con AWS CLI

Sostituisci tutti i *`example values`* nelle fasi di questa procedura con i tuoi valori.

1. Emettere i seguenti comandi per impostare alcune variabili utilizzate nelle fasi successive. Sostituiscilo *`region-code`* con Regione AWS quello in cui desideri distribuire le tue risorse. Il valore può essere qualsiasi Regione AWS valore supportato da Amazon EKS. Per un elenco di Regioni AWS, consulta gli endpoint e le quote di Amazon EKS nella guida di riferimento AWS generale. Sostituisci *`my-cluster`* con un nome da assegnare al cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster. Sostituisci *`my-nodegroup`* con un nome per il gruppo di nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una

lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura. Sostituisci *111122223333* con l'ID del tuo account.

```
export region_code=region-code
export cluster_name=my-cluster
export nodegroup_name=my-nodegroup
export account_id=111122223333
```

2. Creare un Amazon VPC con sottoreti pubbliche e private che soddisfino i requisiti Amazon EKS e IPv6.

   a. Esegui il comando seguente per impostare una variabile per il nome AWS CloudFormation dello stack. Puoi sostituire *my-eks-ipv6-vpc* con un nome a tua scelta.

   ```
   export vpc_stack_name=my-eks-ipv6-vpc
   ```

   b. Crea un IPv6 VPC utilizzando un AWS CloudFormation modello.

   ```
   aws cloudformation create-stack --region $region_code --stack-name
    $vpc_stack_name \
     --template-url https://s3.us-west-2.amazonaws.com/amazon-
   eks/cloudformation/2020-10-29/amazon-eks-ipv6-vpc-public-private-
   subnets.yaml
   ```

   La creazione dello stack richiede alcuni minuti. Esegui il comando seguente. Non passare alla fase successiva finché l'output del comando non è CREATE_COMPLETE.

   ```
   aws cloudformation describe-stacks --region $region_code --stack-name
    $vpc_stack_name --query Stacks[].StackStatus --output text
   ```

   c. Recuperare gli ID delle sottoreti pubbliche che sono state create.

   ```
   aws cloudformation describe-stacks --region $region_code --stack-name
    $vpc_stack_name \
       --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --
   output text
   ```

   Di seguito viene riportato un output di esempio:

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE
```

d. Abilitare l'opzione di assegnazione automatica dell'indirizzo IPv6 per le sottoreti pubbliche create.

```
aws ec2 modify-subnet-attribute --region $region_code --
subnet-id subnet-0a1a56c486EXAMPLE --assign-ipv6-address-on-
creation
aws ec2 modify-subnet-attribute --region $region_code --subnet-id
 subnet-099e6ca77aEXAMPLE --assign-ipv6-address-on-creation
```

e. Recupera i nomi delle sottoreti e dei gruppi di sicurezza creati dal modello dallo AWS CloudFormation stack distribuito e memorizzali in variabili da utilizzare in un passaggio successivo.

```
security_groups=$(aws cloudformation describe-stacks --region $region_code
 --stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SecurityGroups`].OutputValue' --
output text)

public_subnets=$(aws cloudformation describe-stacks --region $region_code --
stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPublic`].OutputValue' --
output text)

private_subnets=$(aws cloudformation describe-stacks --region $region_code
 --stack-name $vpc_stack_name \
    --query='Stacks[].Outputs[?OutputKey==`SubnetsPrivate`].OutputValue' --
output text)

subnets=${public_subnets},${private_subnets}
```

3. Crea un ruolo IAM del cluster e associa la policy gestita Amazon EKS IAM richiesta. Kubernetesi cluster gestiti da Amazon EKS effettuano chiamate ad altri AWS servizi per tuo conto per gestire le risorse che utilizzi con il servizio.

a. Per creare il file `eks-cluster-role-trust-policy.json`, emetti il seguente comando:

```
cat >eks-cluster-role-trust-policy.json <<EOF
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

b. Eseguire questo comando per impostare una variabile per il nome del ruolo. Puoi sostituire *myAmazonEKSClusterRole* con un nome a tua scelta.

```
export cluster_role_name=myAmazonEKSClusterRole
```

c. Crea il ruolo.

```
aws iam create-role --role-name $cluster_role_name --assume-role-policy-
document file://"eks-cluster-role-trust-policy.json"
```

d. Recuperare l'ARN del ruolo IAM e memorizzarlo in una variabile per una fase successiva.

```
cluster_iam_role=$(aws iam get-role --role-name $cluster_role_name --
query="Role.Arn" --output text)
```

e. Allegare la policy IAM gestita da Amazon EKS richiesta al ruolo.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name $cluster_role_name
```

4. Creare il cluster.

```
aws eks create-cluster --region $region_code --name $cluster_name --kubernetes-
version 1.XX \
   --role-arn $cluster_iam_role --resources-vpc-config subnetIds=
$subnets,securityGroupIds=$security_groups \
   --kubernetes-network-config ipFamily=ipv6
```

- > **Note**
  >
  > Potresti ricevere un messaggio di errore indicante che una delle zone di disponibilità nella richiesta non dispone di capacità sufficiente per creare un cluster Amazon EKS. In questo caso, l'output di errore contiene le zone di disponibilità in grado di supportare un nuovo cluster. Riprova a creare il cluster con almeno due sottoreti che si trovano nelle zone di disponibilità supportate per il tuo account. Per ulteriori informazioni, consulta Capacità insufficiente.

  La creazione del cluster richiede diversi minuti. Esegui il comando seguente. Non passare alla fase successiva finché l'output del comando non è `ACTIVE`.

  ```
  aws eks describe-cluster --region $region_code --name $cluster_name --query
   cluster.status
  ```

5. Creare o aggiornare un file `kubeconfig` per il cluster in modo che sia possibile comunicare con il cluster.

   ```
   aws eks update-kubeconfig --region $region_code --name $cluster_name
   ```

   Per impostazione predefinita, il file `config` viene creato in `~/.kube` o la configurazione del nuovo cluster viene aggiunta a un file `config` esistente in `~/.kube`.

6. Creare un ruolo IAM del nodo.

   a. Per creare il file `vpc-cni-ipv6-policy.json`, emetti il seguente comando:

   ```
   cat >vpc-cni-ipv6-policy <<EOF
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "ec2:AssignIpv6Addresses",
                   "ec2:DescribeInstances",
                   "ec2:DescribeTags",
                   "ec2:DescribeNetworkInterfaces",
                   "ec2:DescribeInstanceTypes"
   ```

```
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
EOF
```

b. Creare la policy IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

c. Per creare il file `node-role-trust-relationship.json`, emetti il seguente comando:

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

d. Eseguire questo comando per impostare una variabile per il nome del ruolo. Puoi sostituire *AmazonEKSNodeRole* con un nome a tua scelta.

```
export node_role_name=AmazonEKSNodeRole
```

e. Crea il ruolo IAM.

```
aws iam create-role --role-name $node_role_name --assume-role-policy-
document file://"node-role-trust-relationship.json"
```

f. Allegare la policy IAM al ruolo IAM.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy \
    --role-name $node_role_name
```

> **Important**
>
> Per semplicità, in questo tutorial la policy è allegata a questo ruolo IAM. Tuttavia, in un cluster di produzione, si consiglia di allegare la policy a un ruolo IAM separato. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

g. Allegare al ruolo IAM le due policy gestite IAM richieste.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEC2ContainerRegistryReadOnly \
  --role-name $node_role_name
```

h. Recuperare l'ARN del ruolo IAM e memorizzarlo in una variabile per una fase successiva.

```
node_iam_role=$(aws iam get-role --role-name $node_role_name --
query="Role.Arn" --output text)
```

7. Creare un gruppo di nodi gestito.

a. Visualizzare gli ID delle sottoreti creati in una fase precedente.

```
echo $subnets
```

Di seguito viene riportato un output di esempio:

```
subnet-0a1a56c486EXAMPLE,subnet-099e6ca77aEXAMPLE,subnet-
0377963d69EXAMPLE,subnet-0c05f819d5EXAMPLE
```

b. Creare il gruppo di nodi. Sostituisci *0a1a56c486EXAMPLE*, *099e6ca77aEXAMPLE*, *0377963d69EXAMPLE*, and *0c05f819d5EXAMPLE* con i valori restituiti nell'output della fase precedente. Assicurarsi di rimuovere le virgole tra gli ID delle sottoreti dall'output precedente nel comando seguente. Puoi sostituire *t3.medium* con qualsiasi tipo di istanza di AWS Nitro System.

```
aws eks create-nodegroup --region $region_code --cluster-name $cluster_name
 --nodegroup-name $nodegroup_name \
    --subnets subnet-0a1a56c486EXAMPLE subnet-099e6ca77aEXAMPLE
 subnet-0377963d69EXAMPLE subnet-0c05f819d5EXAMPLE \
    --instance-types t3.medium --node-role $node_iam_role
```

La creazione del gruppo di nodi richiede alcuni minuti. Esegui il comando seguente. Non andare al passaggio successivo finché l'output restituito non è ACTIVE.

```
aws eks describe-nodegroup --region $region_code --cluster-name
 $cluster_name --nodegroup-name $nodegroup_name \
    --query nodegroup.status --output text
```

8. Confermare che ai Pods di default siano assegnati gli indirizzi IPv6 nella colonna IP.

```
kubectl get pods -n kube-system -o wide
```

Di seguito viene riportato un output di esempio:

```
NAME                       READY   STATUS    RESTARTS   AGE     IP
                             NODE
 NOMINATED NODE   READINESS GATES
aws-node-rslts             1/1     Running   1          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
```

```
aws-node-t74jh             1/1     Running   0          5m32s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-cw7w2   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::                ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
coredns-85d5b4454c-tx6n8   1/1     Running   0          56m
  2600:1f13:b66:8203:34e5::1               ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
kube-proxy-btpbk           1/1     Running   0          5m36s
  2600:1f13:b66:8200:11a5:ade0:c590:6ac8   ip-192-168-34-75.region-
code.compute.internal   <none>           <none>
kube-proxy-jjk2g           1/1     Running   0          5m33s
  2600:1f13:b66:8203:4516:2080:8ced:1ca9   ip-192-168-253-70.region-
code.compute.internal  <none>           <none>
```

9. Confermare che ai servizi di default siano assegnati gli indirizzi IPv6 nella colonna IP.

   ```
   kubectl get services -n kube-system -o wide
   ```

   Di seguito viene riportato un output di esempio:

   ```
   NAME       TYPE        CLUSTER-IP          EXTERNAL-IP   PORT(S)         AGE
    SELECTOR
   kube-dns   ClusterIP   fd30:3087:b6c2::a   <none>        53/UDP,53/TCP   57m
    k8s-app=kube-dns
   ```

10. (Facoltativo) Implementa un'applicazione di esempio o implementa il AWS Load Balancer Controller e un'applicazione di esempio per eseguire il bilanciamento del carico dell'applicazione o del traffico di rete verso i Pods IPv6.
11. Una volta terminato con il cluster e i nodi creati per questo tutorial, è necessario eliminare le risorse create con i seguenti comandi. Assicurarsi di non utilizzare nessuna delle risorse al di fuori di questo tutorial prima di eliminarle.

    a. Se stai completando questo passaggio in una shell diversa da quella in cui sono stati completati i passaggi precedenti, imposta i valori di tutte le variabili utilizzate nei passaggi precedenti, sostituendo i *example values* con i valori specificati durante i passaggi precedenti. Se si sta completando questo passaggio nella stessa shell in cui sono stati completati i passaggi precedenti, andare alla fase successiva.

       ```
       export region_code=region-code
       ```

```
export vpc_stack_name=my-eks-ipv6-vpc
export cluster_name=my-cluster
export nodegroup_name=my-nodegroup
export account_id=111122223333
export node_role_name=AmazonEKSNodeRole
export cluster_role_name=myAmazonEKSClusterRole
```

b. Eliminare il gruppo di nodi.

```
aws eks delete-nodegroup --region $region_code --cluster-name $cluster_name
 --nodegroup-name $nodegroup_name
```

L'eliminazione richiede pochi minuti. Esegui il comando seguente. Non andare al passaggio successivo se viene restituito un output.

```
aws eks list-nodegroups --region $region_code --cluster-name $cluster_name
 --query nodegroups --output text
```

c. Elimina il cluster.

```
aws eks delete-cluster --region $region_code --name $cluster_name
```

L'eliminazione del cluster richiede alcuni minuti. Prima di continuare, accertarsi che il cluster sia eliminato con il comando seguente:

```
aws eks describe-cluster --region $region_code --name $cluster_name
```

Non andare al passaggio successivo finché l'output non è simile a quello seguente:

```
An error occurred (ResourceNotFoundException) when calling the
 DescribeCluster operation: No cluster found for name: my-cluster.
```

d. Eliminare le risorse IAM create. Sostituisci *AmazonEKS_CNI_IPv6_Policy* con il nome scelto, se è stato scelto un nome diverso da quello usato nei passaggi precedenti.

```
aws iam detach-role-policy --role-name $cluster_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
```

```
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
aws iam detach-role-policy --role-name $node_role_name --policy-arn
 arn:aws:iam::$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-policy --policy-arn arn:aws:iam::
$account_id:policy/AmazonEKS_CNI_IPv6_Policy
aws iam delete-role --role-name $cluster_role_name
aws iam delete-role --role-name $node_role_name
```

e. Elimina lo AWS CloudFormation stack che ha creato il VPC.

```
aws cloudformation delete-stack --region $region_code --stack-name
 $vpc_stack_name
```

SNAT per Pods

Le informazioni contenute in questa sezione non sono applicabili al cluster se la sua implementazione è avvenuta tramite la famiglia `IPv6`, in quanto gli indirizzi `IPv6` non sono tradotti in rete. Per ulteriori informazioni sull'uso di `IPv6` con il cluster, consulta IPv6indirizzi per cluster Pods e services.

Per impostazione predefinita, a ciascun Pod nel cluster viene assegnato un indirizzo `IPv4` privato da un instradamento interdominio senza classi (CIDR) associato al VPC in cui è implementato il Pod. I Pods nello stesso VPC comunicano tra loro utilizzando questi indirizzi IP privati come endpoint. Per impostazione predefinita, quando un Pod comunica a un indirizzo `IPv4` che non è incluso in un blocco CIDR associato al VPC, il plug-in CNI di Amazon VPC (per Linux oppure Windows) traduce l'indirizzo `IPv4` del Pod's nell'indirizzo `IPv4` privato principale dell'interfaccia di rete elastica principale del nodo su cui è in esecuzione il Pod*.

> **Note**
>
> Per i nodi Windows, ci sono ulteriori dettagli da considerare. Per impostazione predefinita, il plug-in CNI VPC per Windows è definito con una configurazione di rete in cui il traffico verso una destinazione all'interno dello stesso VPC è escluso per SNAT. Ciò significa che la comunicazione VPC interna ha SNAT disabilitato e l'indirizzo IP assegnato al Pod è indirizzabile all'interno del VPC. Tuttavia, il traffico verso una destinazione esterna al VPC ha l'IP Pod di origine SNAT associato all'indirizzo IP primario dell'interfaccia di rete elastica (ENI) dell'istanza. Questa configurazione predefinita per Windows garantisce che il pod possa accedere alle reti esterne al VPC allo stesso modo dell'istanza host.

A causa di questo comportamento:

- I Pods possono comunicare in modo bidirezionale con le risorse Internet solo se il nodo su cui sono in esecuzione dispone di un indirizzo IP pubblico o elastico ad esso assegnato e si trova in una sottorete pubblica. Alle sottoreti pubbliche è associata una tabella di instradamento con un instradamento a un gateway Internet. Per questo motivo, ti consigliamo di implementare i nodi nelle sottoreti private, quando possibile.
- Per le versioni del plugin precedenti alla `1.8.0`, le risorse che si trovano nelle reti o nei VPC collegati al VPC del cluster tramite peering VPC, un VPC di transito o i AWS Direct Connect non possono iniziare una comunicazione con i Pods oltre le interfacce di rete elastiche secondarie. I Pods possono tuttavia comunicare con tali risorse e ricevere risposte.

Se una delle seguenti affermazioni è vera nel tuo ambiente, modifica la configurazione predefinita con il comando che segue.

- È necessario avere risorse nelle reti o nei VPC collegati al VPC del cluster tramite peering VPC, un VPC di transito o AWS Direct Connect che devono iniziare la comunicazione con i Pods tramite un indirizzo `IPv4`, con la versione del plugin precedente alla `1.8.0`.
- I tuoi Pods si trovano in una sottorete privata e devono comunicare in uscita a Internet. La sottorete dispone di una route a un gateway NAT.

```
kubectl set env daemonset -n kube-system aws-node AWS_VPC_K8S_CNI_EXTERNALSNAT=true
```

> **Note**
>
> Le variabili di configurazione `AWS_VPC_K8S_CNI_EXTERNALSNAT` e `AWS_VPC_K8S_CNI_EXCLUDE_SNAT_CIDRS` di CNI non sono applicabili ai nodi Windows. La disabilitazione di SNAT non è supportata per Windows. Per quanto riguarda l'esclusione di un elenco di CIDR `IPv4` da SNAT, è possibile definirla specificando il parametro `ExcludedSnatCIDRs` nello script di bootstrap di Windows. Per ulteriori informazioni su questo parametro, consulta la sezione Parametri di configurazione dello script di bootstrap.

*Se la specifica di un Pod's contiene `hostNetwork=true` (il valore predefinito è `false`), il suo indirizzo IP non viene convertito in un indirizzo diverso. Per impostazione predefinita, questo comportamento è presente per i Pods di `kube-proxy` e Amazon VPC CNI plugin

for Kubernetes in esecuzione sul cluster. Per questi Pods, l'indirizzo IP corrisponde a quello principale del nodo, poiché l'indirizzo IP del Pod's non viene convertito. Per ulteriori informazioni su un'Pod's`hostNetwork`impostazione, consulta PodSpec v1 core nel riferimento Kubernetes API.

Configurazione del cluster per le policy di rete Kubernetes

Per impostazione predefinita, in Kubernetes non ci sono restrizioni per indirizzi IP, porte o connessioni tra qualsiasi Pods nel cluster o tra i tuoi Pods e le risorse in qualsiasi altra rete. È possibile utilizzare le policy di rete Kubernetes per limitare il traffico di rete da e verso i Pods. Per ulteriori informazioni, consulta la pagina Network Policies nella documentazione di Kubernetes.

Se hai una versione `1.13` o precedente del Amazon VPC CNI plugin for Kubernetes sul cluster, è necessario implementare una soluzione di terze parti per applicare le policy di rete Kubernetes al cluster. Una versione `1.14` o successiva del plugin è in grado di implementare le policy di rete, quindi non è necessario utilizzare una soluzione di terze parti. In questo argomento, imparerai a configurare il cluster per l'utilizzo delle policy di rete Kubernetes sul cluster senza utilizzare componenti aggiuntivi di terze parti.

Le policy di rete nel Amazon VPC CNI plugin for Kubernetes sono supportate nelle seguenti configurazioni.

- Cluster Amazon EKS versione `1.25` e successive.
- Versione 1.14 o successiva del Amazon VPC CNI plugin for Kubernetes sul cluster.
- Cluster configurato per indirizzi `IPv4` o `IPv6`.
- È possibile utilizzare le policy di rete con i gruppi di sicurezza per Pods. Con le policy di rete, è possibile controllare tutte le comunicazioni all'interno del cluster. Con i gruppi di sicurezza perPods, puoi controllare l'accesso Servizi AWS alle applicazioni all'interno di unPod.
- È possibile utilizzare le policy di rete con le reti personalizzate e la delega del prefisso.

Considerazioni

- Quando si applicano le policy di rete Amazon VPC CNI plugin for Kubernetes per il cluster con il Amazon VPC CNI plugin for Kubernetes, è possibile applicare le policy solo ai nodi Amazon EC2 Linux. Non è possibile applicare le policy ai nodi Fargate o Windows.
- Se il cluster utilizza attualmente una soluzione di terze parti per la gestione delle policy di rete Kubernetes, è possibile utilizzare le stesse policy con il Amazon VPC CNI plugin for Kubernetes. Tuttavia, è necessario rimuovere la soluzione esistente in modo che non gestisca le stesse policy.

- È possibile applicare più policy di rete allo stesso Pod. Quando sono configurate due o più policy che selezionano lo stesso Pod, al Pod vengono applicate tutte le policy.
- Il numero massimo di combinazioni univoche di porte per ogni protocollo in ogni `ingress:` `egress:` selettore in una politica di rete è 24.
- Per ognuno dei servizi Kubernetes, la porta del servizio deve essere uguale a quella del container. Se si utilizzano porte con un nome assegnato, è necessario utilizzare lo stesso nome anche nelle specifiche del servizio.
- Applicazione delle policy all'avvio Pod

  Il Amazon VPC CNI plugin for Kubernetes configura le policy di rete per i pod parallelamente al provisioning dei pod. Fino a quando tutte le policy non saranno configurate per il nuovo pod, i contenitori nel nuovo pod inizieranno con una policy di autorizzazione predefinita. Questa è chiamata modalità standard. Una politica di autorizzazione predefinita significa che tutto il traffico in entrata e in uscita è consentito da e verso i nuovi pod.

  È possibile modificare questa politica di rete predefinita impostando la variabile `NETWORK_POLICY_ENFORCING_MODE` di ambiente su `strict` nel `aws-node` contenitore del CNI VPC. `DaemonSet`

  ```
  env:
    - name: NETWORK_POLICY_ENFORCING_MODE
      value: "strict"
  ```

  Con la `NETWORK_POLICY_ENFORCING_MODE` variabile impostata su`strict`, i pod che utilizzano il CNI VPC iniziano con una politica di negazione predefinita, quindi vengono configurate le politiche. Questa è chiamata modalità rigorosa. In modalità rigorosa, è necessario disporre di una politica di rete per ogni endpoint a cui i pod devono accedere nel cluster. Tieni presente che questo requisito si applica ai pod. CoreDNS La politica di negazione predefinita non è configurata per i pod con rete host.
- La funzionalità di policy di rete crea e richiede una `PolicyEndpoint` Custom Resource Definition (CRD) denominata `policyendpoints.networking.k8s.aws`. Gli oggetti `PolicyEndpoint` della Custom Resource sono gestiti da Amazon EKS. È sconsigliabile modificare o eliminare queste risorse.
- Se si eseguono pod che utilizzano le credenziali IAM del ruolo di istanza o si connettono all'IMDS EC2, occorre accertarsi che non siano presenti policy di rete che bloccherebbero l'accesso all'IMDS EC2. Potrebbe essere necessario aggiungere una policy di rete per consentire l'accesso

all'IMDS EC2. Per ulteriori informazioni, consulta Metadati dell'istanza e dati dell'utente nella Guida per l'utente di Amazon EC2 per le istanze Linux.

I pod che utilizzano i ruoli IAM per gli account di servizio non accedono all'IMDS EC2.

- L'Amazon VPC CNI plugin for Kubernetes non applica le policy di rete alle interfacce di rete aggiuntive per ogni pod, ma solo all'interfaccia principale per ogni pod (`eth0`). Ciò influisce sulle seguenti architetture:
  - Pod `IPv6` con la variabile `ENABLE_V4_EGRESS` impostata su `true`. Questa variabile abilita la funzionalità di uscita `IPv4` per connettere i pod IPv6 a endpoint `IPv4` come quelli esterni al cluster. La funzionalità di uscita `IPv4` opera creando un'interfaccia di rete supplementare con un indirizzo IPv4 di loopback locale.
  - Quando si utilizzano plugin di rete concatenati come. Multus Poiché questi plugin aggiungono interfacce di rete a ciascun pod, le policy di rete non vengono applicate ai plug-in di rete concatenati.
- Per impostazione predefinita, per i parametri la funzionalità di policy di rete utilizza la porta `8162` sul nodo. Inoltre, la funzionalità utilizzava la porta `8163` per le sonde dell'integrità. Se esegui un'altra applicazione sui nodi o all'interno dei pod che deve utilizzare queste porte, l'app non viene eseguita. Nella versione VPC CNI `v1.14.1` o successiva, puoi modificare la porta di queste porte nelle seguenti posizioni:

  AWS Management Console

  1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
  2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui desideri configurare il componente aggiuntivo CNI di Amazon VPC.
  3. Seleziona la scheda Componenti aggiuntivi.
  4. Seleziona la casella nella parte superiore destra della casella del componente aggiuntivo e scegli Edit (Modifica).
  5. Nella pagina Configure ***`name of addon`*** (Configura il nome del componente aggiuntivo):
     a. Seleziona una `v1.14.0-eksbuild.3` o versione successiva nell'elenco a discesa Versione.
     b. Scegli Impostazioni di configurazione facoltative.
     c. Inserisci la chiave JSON `"enableNetworkPolicy":` e il valore `"true"` nei Valori di configurazione. Il testo risultante deve essere un oggetto JSON valido. Se questa

chiave e questo valore sono gli unici dati nella casella di testo, racchiudi la chiave e il valore tra parentesi graffe `{}`.

L'esempio seguente ha la funzionalità delle politiche di rete abilitate, i log delle politiche di rete abilitati, i log delle politiche di rete inviati ad Amazon CloudWatch Logs e le metriche e i probe di salute sono impostati sui numeri di porta predefiniti:

```
{
    "enableNetworkPolicy": "true",
    "nodeAgent": {
        "enablePolicyEventLogs": "true",
        "enableCloudWatchLogs": "true",
        "healthProbeBindAddr": "8163",
        "metricsBindAddr": "8162"
    }
}
```

Helm

Se il Amazon VPC CNI plugin for Kubernetes è stato installato attraverso `helm`, è possibile aggiornare la configurazione per abilitare le porte.

- Esegui il seguente comando per modificare le porte. Imposta il numero di porta nel valore, rispettivamente, della chiave `nodeAgent.metricsBindAddr` o `nodeAgent.healthProbeBindAddr`.

```
helm upgrade --set nodeAgent.metricsBindAddr=8162 --set
 nodeAgent.healthProbeBindAddr=8163 aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. Apri la `DaemonSet` del `aws-node` nell'editor.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Sostituisci i numeri di porta negli argomenti del comando seguente nel `args:` del container `aws-network-policy-agent` nel manifesto del daemonset `aws-node` della CNI del VPC.

```
    - args:
            - --metrics-bind-addr=:8162
            - --health-probe-bind-addr=:8163
```

Prerequisiti

- Versione minima del cluster

  Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS. Il cluster deve essere Kubernetes versione `1.25` o successiva. Il cluster deve eseguire una delle versioni Kubernetes e della piattaforma elencate nella tabella seguente. Sono supportate anche tutte le versioni di Kubernetes e della piattaforma successive a quelle elencate. È possibile verificare la versione corrente di Kubernetes sostituendo *`my-cluster`* nel seguente comando con il nome del cluster e quindi eseguendo il comando modificato:

```
aws eks describe-cluster
              --name my-cluster --query cluster.version --output
              text
```

Versione Kubernetes	Versione della piattaforma
`1.27.4`	`eks.5`
`1.26.7`	`eks.6`
`1.25.12`	`eks.7`

- Versione minima VPC CNI

  Versione `1.14` o successiva del Amazon VPC CNI plugin for Kubernetes sul cluster. È possibile verificare la versione attuale con il seguente comando.

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni: |
 cut -d : -f 3
```

Se la versione è precedente alla `1.14`, consulta la pagina Aggiornamento del componente aggiuntivo di Amazon EKS per eseguire l'aggiornamento alla versione `1.14` o successiva.

- Versione minima del kernel Linux

  I nodi devono avere una versione del kernel Linux `5.10` o successiva. È possibile verificare la versione del kernel con `uname -r`. Se utilizzi le versioni più recenti delle AMI Amazon Linux ottimizzate per Amazon EKS, Amazon Linux accelerate ottimizzate per Amazon EKS e Bottlerocket, queste dispongono già della versione del kernel richiesta.

  La versione `v20231116` o successive delle AMI Amazon Linux accelerate ottimizzate per Amazon EKS dispone della versione del kernel `5.10`.

Per configurare il cluster per l'utilizzo delle policy di rete Kubernetes

1. Installazione del file system BPF

   > **Note**
   >
   > Se il cluster è una versione `1.27` o successiva, è possibile saltare questo passaggio poiché tutte le AMI Amazon Linux e Bottlerocket ottimizzate per Amazon EKS per `1.27` o versioni successive dispongono già di questa funzionalità.
   > Per tutte le altre versioni del cluster, se si esegue l'upgrade di Amazon Linux ottimizzato per Amazon EKS alla versione `v20230703` o successiva oppure si aggiorna l'AMI Bottlerocket alla versione `v1.0.2` o successiva, è possibile saltare questo passaggio.

   a. Installa il file system Berkeley Packet Filter (BPF) su ciascuno dei nodi.

      ```
      sudo mount -t bpf bpffs /sys/fs/bpf
      ```

   b. Quindi, aggiungi lo stesso comando ai dati utente nel modello di avvio per i gruppi con dimensionamento automatico Amazon EC2.

2. Abilitazione della policy di rete nella CNI di VPC

   a. Scopri qual è il tipo di componente aggiuntivo attualmente installato sul cluster. A seconda dello strumento con cui hai creato il cluster, al momento potresti non avere il componente

aggiuntivo del tipo Amazon EKS installato sul cluster. Sostituisci *my-cluster* con il nome del cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
 addon.addonVersion --output text
```

Se viene restituito il numero di versione, sul cluster è installato il componente aggiuntivo del tipo Amazon EKS e non è necessario completare i passaggi rimanenti di questa procedura. Se viene restituito un errore, sul cluster non è installato il componente aggiuntivo del tipo Amazon EKS.

b. • Componenti aggiuntivi di Amazon EKS

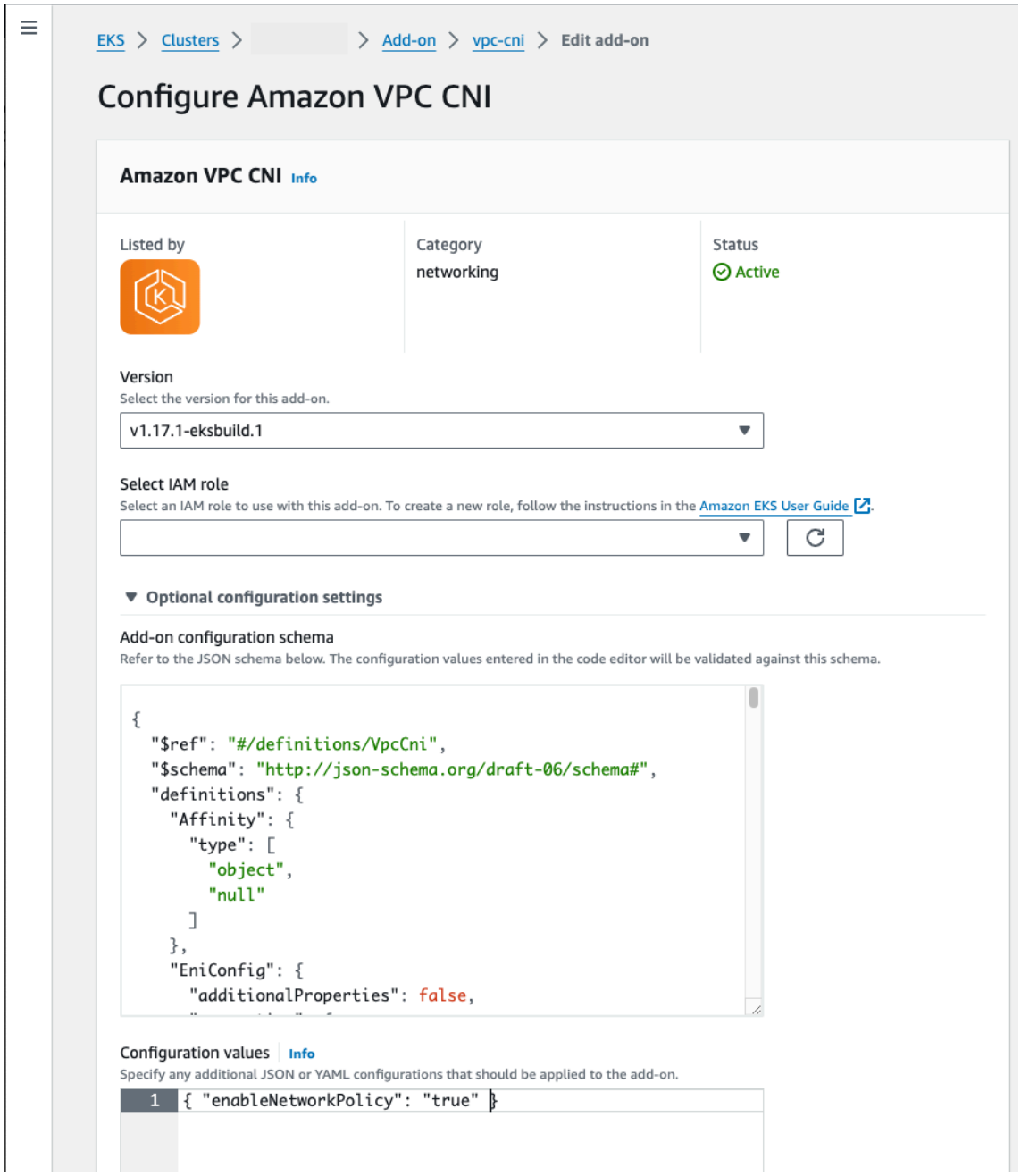
AWS Management Console

a. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
b. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui desideri configurare il componente aggiuntivo CNI di Amazon VPC.
c. Seleziona la scheda Componenti aggiuntivi.
d. Seleziona la casella nella parte superiore destra della casella del componente aggiuntivo e scegli Edit (Modifica).
e. Nella pagina Configure ***name of addon*** (Configura il nome del componente aggiuntivo):

i. Seleziona una `v1.14.0-eksbuild.3` o versione successiva nell'elenco a discesa Versione.
ii. Scegli Impostazioni di configurazione facoltative.
iii. Inserisci la chiave JSON `"enableNetworkPolicy":` e il valore `"true"` nei Valori di configurazione. Il testo risultante deve essere un oggetto JSON valido. Se questa chiave e questo valore sono gli unici dati nella casella di testo, racchiudi la chiave e il valore tra parentesi graffe `{}`. L'esempio seguente mostra che la policy di rete è abilitata:

```
{ "enableNetworkPolicy": "true" }
```

Lo screenshot seguente mostra un esempio di tale scenario.



AWS CLI

- Esegui il comando seguente. AWS CLI Sostituisci `my-cluster` con il nome del cluster e l'ARN del ruolo IAM con il ruolo che stai utilizzando.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni
 --addon-version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values
 '{"enableNetworkPolicy": "true"}'
```

- Componenti aggiuntivi autogestiti

  Helm

  Se il Amazon VPC CNI plugin for Kubernetes è stato installato attraverso `helm`, è possibile aggiornare la configurazione per abilitare la policy di rete.

  - Esegui il comando seguente per abilitare la policy di rete.

    ```
    helm upgrade --set enableNetworkPolicy=true aws-vpc-cni --namespace
     kube-system eks/aws-vpc-cni
    ```

  kubectl

  a. Apri la `ConfigMap` del `amazon-vpc-cni` nell'editor.

     ```
     kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
     ```

  b. Aggiungi la seguente riga ai `data` nella `ConfigMap`.

     ```
     enable-network-policy-controller: "true"
     ```

     Dopo aver aggiunto la riga, la `ConfigMap` dovrebbe essere simile al seguente esempio.

     ```
     apiVersion: v1
      kind: ConfigMap
      metadata:
       name: amazon-vpc-cni
       namespace: kube-system
      data:
     ```

```
  enable-network-policy-controller: "true"
```

c. Apri la `DaemonSet` del `aws-node` nell'editor.

```
kubectl edit daemonset -n kube-system aws-node
```

d. Sostituisci il valore `false` con `true` nell'argomento del comando `--enable-network-policy=false` in `args:` nel container `aws-network-policy-agent` nel manifesto del daemonset `aws-node` della CNI di VPC.

```
    - args:
       - --enable-network-policy=true
```

3. Verifica che i pod `aws-node` siano in esecuzione sul cluster.

```
kubectl get pods -n kube-system | grep 'aws-node\|amazon'
```

Di seguito viene riportato un output di esempio:

```
aws-node-gmqp7                                          2/2     Running   1 (24h
 ago)   24h
aws-node-prnsh                                          2/2     Running   1 (24h
 ago)   24h
```

Se la policy di rete è abilitata, nei pod `aws-node` sono presenti due container. Nelle versioni precedenti e se le policy di rete sono disabilitate, nei pod `aws-node` è presente un solo container.

Ora è possibile implementare le policy di rete Kubernetes per il cluster. Per ulteriori informazioni, consulta Policy di rete Kubernetes.

Demo delle policy di rete con Stars

La demo consente di creare un servizio di front-end, di back-end e client sul cluster Amazon EKS. Inoltre, la demo consente di creare un'interfaccia utente grafica di gestione che mostra i percorsi di ingresso e uscita disponibili tra ogni servizio. Ti consigliamo di completare la demo su un cluster su cui non esegui carichi di lavoro di produzione.

Prima di creare eventuali policy di rete, tutti i servizi sono in grado di comunicare in modo bidirezionale. Dopo l'applicazione delle policy di rete, puoi vedere che il client può comunicare solo con il servizio di front-end, mentre il back-end accetta traffico solo dal front-end.

Per eseguire la demo policy Stars

1. Applica i servizi di front-end, back-end, client e dell'interfaccia utente di gestione:

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/namespace.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/management-ui.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/backend.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/frontend.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/client.yaml
```

2. Visualizza tutti i Pods nel cluster.

```
kubectl get pods -A
```

Di seguito viene riportato un output di esempio:

Nell'output, dovresti vedere i pod negli spazi dei nomi mostrati nel seguente output. I *NOMI* dei pod e il numero di pod nella colonna READY sono diversi da quelli del seguente output. Non continuare finché non vedi pod con nomi simili e tutti hanno il valore Running nella colonna STATUS.

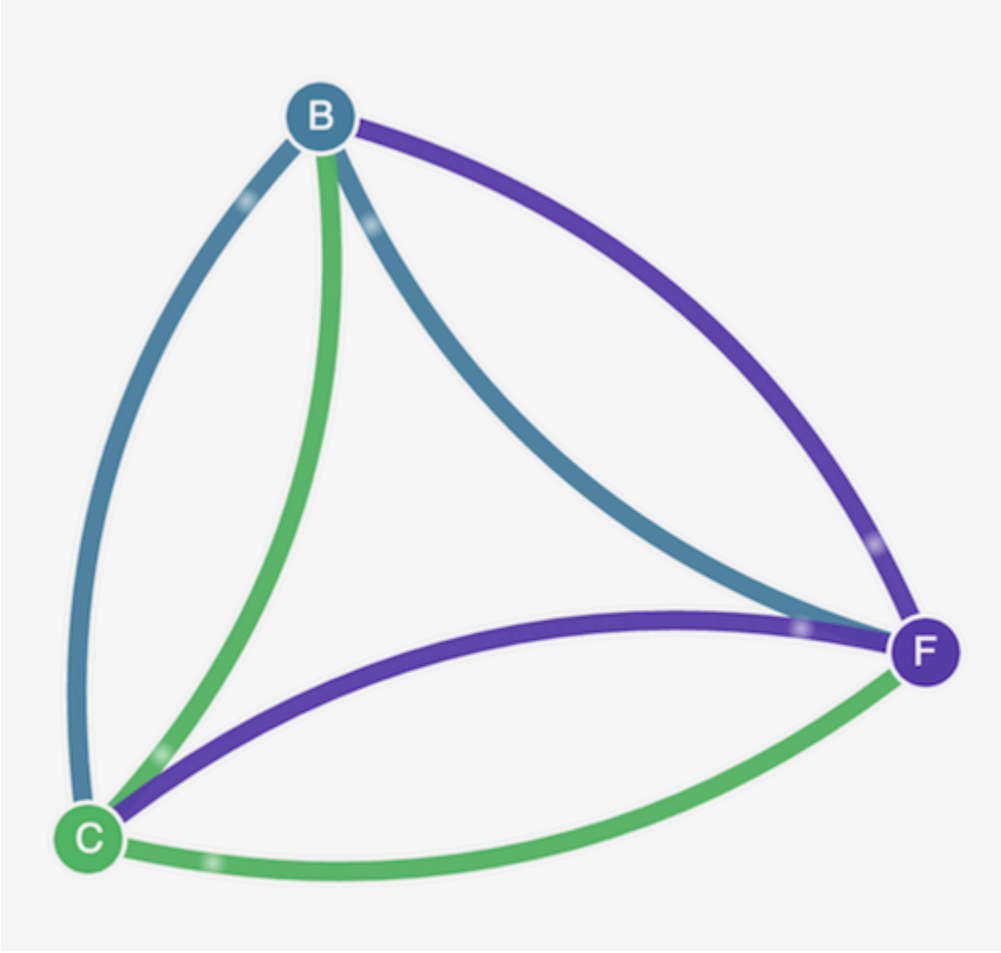
```
NAMESPACE         NAME                                       READY   STATUS
 RESTARTS   AGE
[...]
client            client-xlffc                               1/1     Running   0
       5m19s
[...]
management-ui     management-ui-qrb2g                        1/1     Running   0
       5m24s
stars             backend-sz87q                              1/1     Running   0
       5m23s
```

```
stars              frontend-cscnf                              1/1     Running   0
       5m21s
[...]
```

3. Per connetterti all'interfaccia utente di gestione, connettiti all'`EXTERNAL-IP` del servizio in esecuzione sul cluster:

```
kubectl get service/management-ui -n management-ui
```

4. Apri un browser nella posizione del passaggio precedente. Dovresti visualizzare l'interfaccia utente di gestione seguente. Il nodo C è il servizio client, il nodo F è il servizio front-end e il nodo B è il servizio di back-end. Ogni nodo dispone di accesso di comunicazione completo a tutti gli altri nodi, come indicato dalle righe colorate in grassetto.



5. Applica la seguente policy di rete nei due spazi dei nomi `stars` e `client` per isolare tra loro i servizi:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
```

```
metadata:
  name: default-deny
spec:
  podSelector:
    matchLabels: {}
```

È possibile utilizzare i seguenti comandi per applicare la policy a entrambi gli spazi dei nomi:

```
kubectl apply -n stars -f https://eksworkshop.com/beginner/120_network-policies/
calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
kubectl apply -n client -f https://eksworkshop.com/beginner/120_network-policies/
calico/stars_policy_demo/apply_network_policies.files/default-deny.yaml
```

6. Aggiorna il tuo browser. Poiché l'interfaccia utente di gestione non può più raggiungere i nodi, essi non vengono visualizzati nell'interfaccia utente.
7. Applica le seguenti policy di rete per consentire all'interfaccia utente di gestione di accedere ai servizi. Applica questa policy per consentire all'interfaccia utente di:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: allow-ui
spec:
  podSelector:
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

Applica questa policy per consentire al client di:

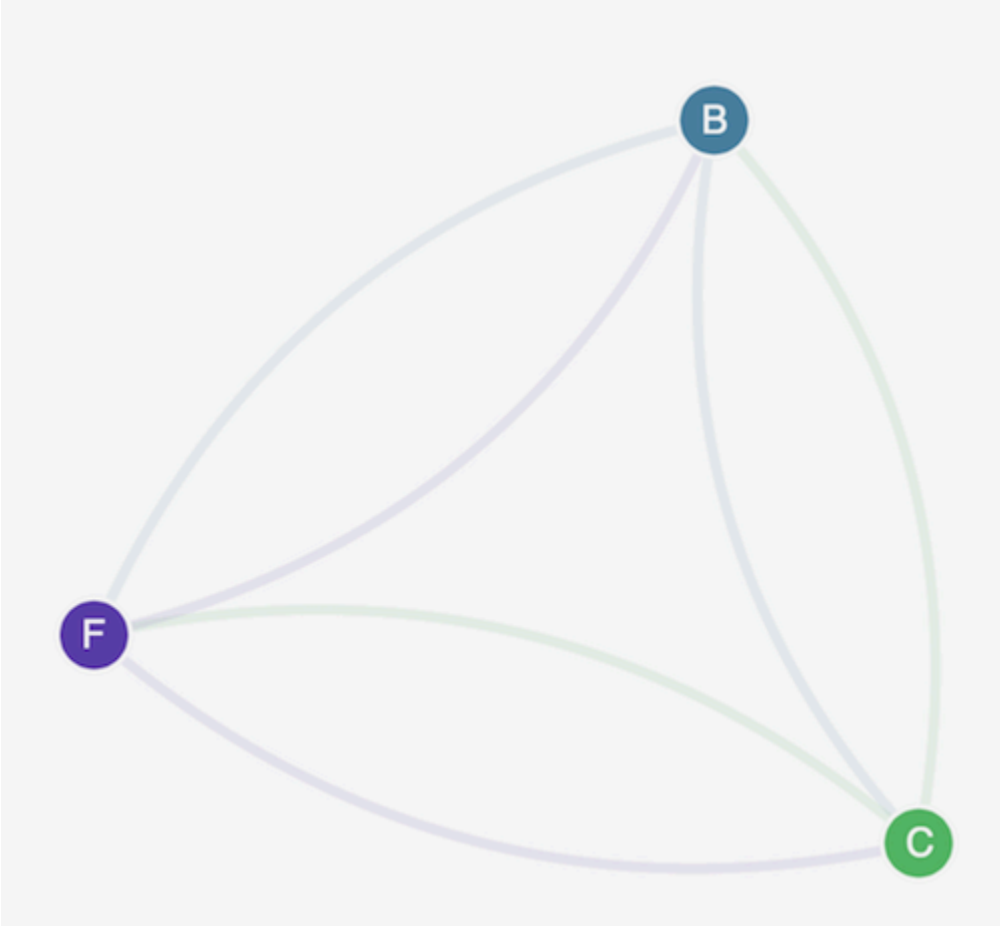
```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: client
  name: allow-ui
spec:
  podSelector:
```

```
    matchLabels: {}
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: management-ui
```

È possibile utilizzare i seguenti comandi per applicare la policy a entrambe le policy:

```
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/apply_network_policies.files/allow-ui.yaml
kubectl apply -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/apply_network_policies.files/allow-ui-client.yaml
```

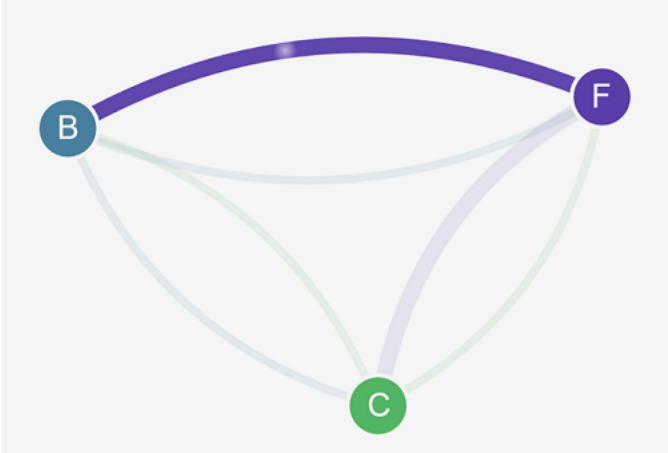
8. Aggiorna il tuo browser. L'interfaccia utente di gestione può raggiungere nuovamente i nodi, ma i nodi non sono in grado di comunicare tra loro.



9. Applicare le seguenti policy di rete per consentire il traffico dal servizio front-end al servizio back-end:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: backend-policy
spec:
  podSelector:
    matchLabels:
      role: backend
  ingress:
    - from:
        - podSelector:
            matchLabels:
              role: frontend
      ports:
        - protocol: TCP
          port: 6379
```

10. Aggiorna il tuo browser. Puoi vedere che il front-end può comunicare con il back-end.

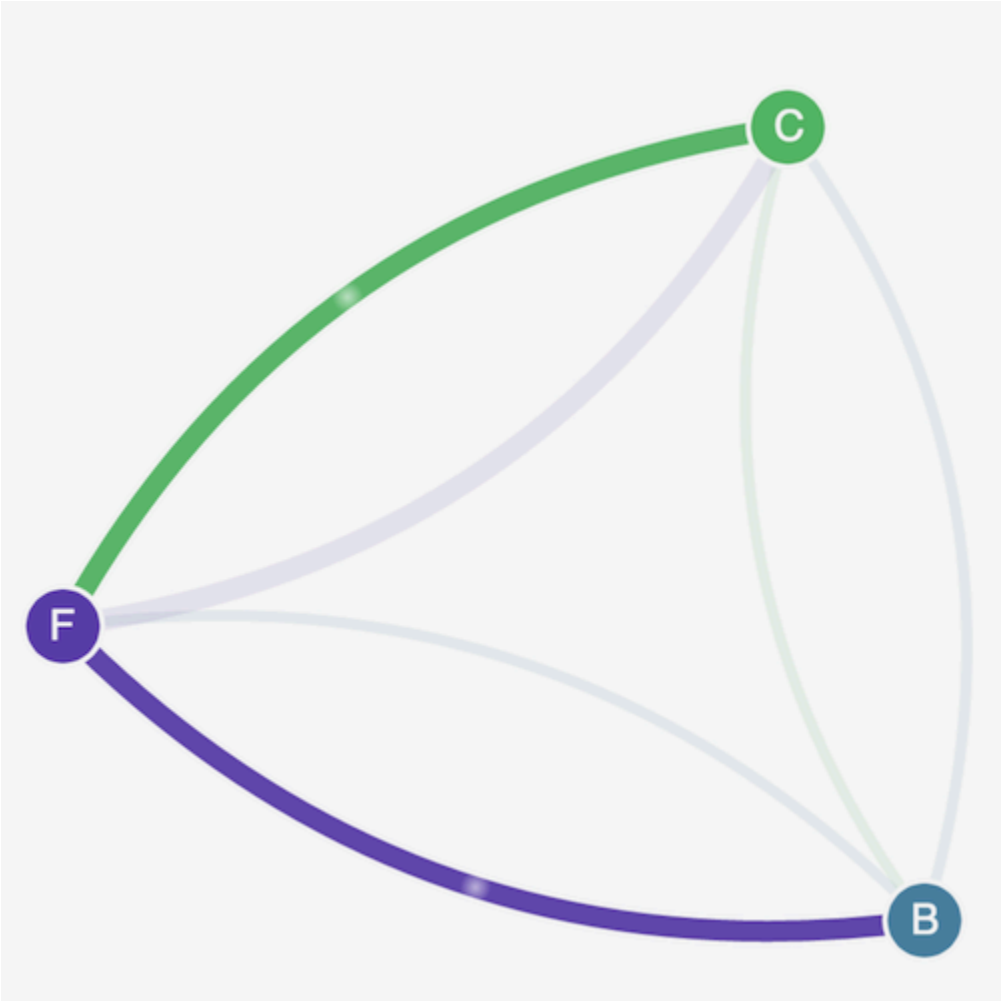


11. Applica la seguente policy di rete per consentire il traffico dal client al servizio di front-end:

```
kind: NetworkPolicy
apiVersion: networking.k8s.io/v1
metadata:
  namespace: stars
  name: frontend-policy
spec:
  podSelector:
    matchLabels:
      role: frontend
```

```
  ingress:
    - from:
        - namespaceSelector:
            matchLabels:
              role: client
      ports:
        - protocol: TCP
          port: 80
```

12. Aggiorna il tuo browser. Puoi vedere che il client può comunicare con il servizio di front-end. Il servizio di front-end può ancora comunicare con il servizio di back-end.



13. (Facoltativo) Al termine della demo, puoi eliminare le relative risorse.

```
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/client.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/frontend.yaml
```

```
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/backend.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/management-ui.yaml
kubectl delete -f https://eksworkshop.com/beginner/120_network-policies/calico/
stars_policy_demo/create_resources.files/namespace.yaml
```

Anche dopo l'eliminazione delle risorse, sui nodi possono essere presenti endpoint di policy di rete che potrebbero interferire in modo imprevisto con la rete nel cluster. L'unico modo sicuro per rimuovere queste regole è riavviare i nodi o terminare tutti i nodi e riciclarli. Per terminare tutti i nodi, impostare il conteggio desiderato del Gruppo Auto Scaling su 0, quindi eseguire il backup al numero desiderato oppure terminare semplicemente i nodi.

Risoluzione dei problemi relativi alle policy di rete

È possibile risolvere i problemi e analizzare le connessioni di rete che utilizzano le policy di rete leggendo i Log delle policy di rete ed eseguendo gli strumenti dell'SDK eBPF.

Log delle policy di rete

I log di flusso registrano se le connessioni sono consentite o negate da una policy di rete. I log delle policy di rete su ogni nodo includono i log di flusso per ogni pod che dispone di una policy di rete. I log delle policy di rete sono archiviati in `/var/log/aws-routed-eni/network-policy-agent.log`. Di seguito è riportato un esempio del file `network-policy-agent.log`:

```
{"level":"info","timestamp":"2023-05-30T16:05:32.573Z","logger":"ebpf-
client","msg":"Flow Info: ","Src
IP":"192.168.87.155","Src Port":38971,"Dest IP":"64.6.160","Dest
Port":53,"Proto":"UDP","Verdict":"ACCEPT"}
```

I registri delle politiche di rete sono disabilitati per impostazione predefinita. Per abilitare i registri delle politiche di rete, procedi nel seguente modo:

> **Note**
>
> I log delle policy di rete richiedono 1 vCPU aggiuntiva per `aws-network-policy-agent` il contenitore nel manifesto del daemonset `aws-node` VPC CNI.

Componenti aggiuntivi di Amazon EKS

AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui desideri configurare il componente aggiuntivo CNI di Amazon VPC.
3. Seleziona la scheda Componenti aggiuntivi.
4. Seleziona la casella nella parte superiore destra della casella del componente aggiuntivo e scegli Edit (Modifica).
5. Nella pagina Configure ***name of addon*** (Configura il nome del componente aggiuntivo):

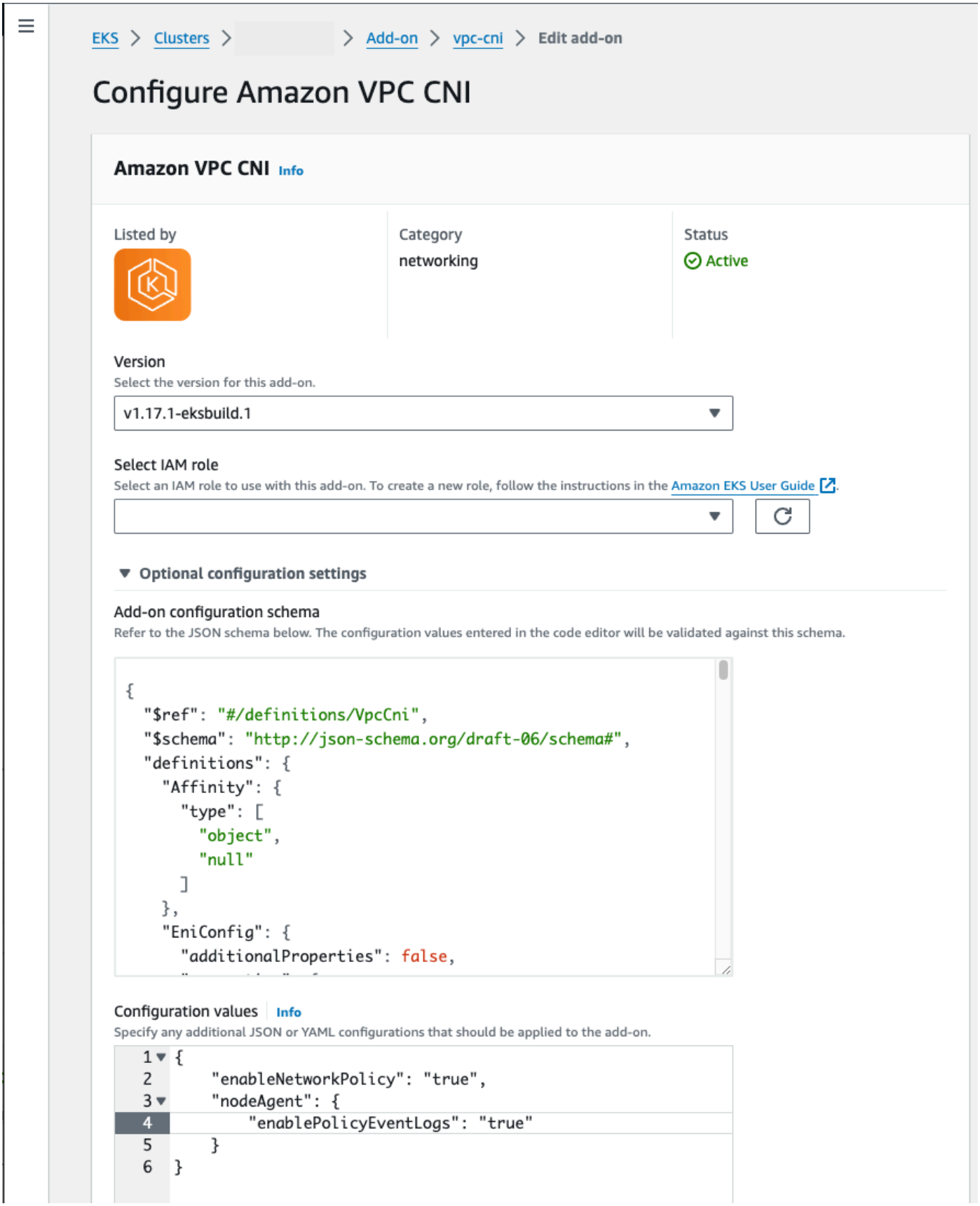
   a. Seleziona una `v1.14.0-eksbuild.3` o versione successiva nell'elenco a discesa Versione.

   b. Scegli Impostazioni di configurazione facoltative.

   c. Inserisci la chiave JSON di primo livello `"nodeAgent":` e il valore, che è un oggetto con una chiave`"enablePolicyEventLogs":` e un valore di `"true"` nei Valori di configurazione. Il testo risultante deve essere un oggetto JSON valido. L'esempio seguente mostra i criteri di rete e i registri dei criteri di rete sono abilitati e i registri dei criteri di rete vengono inviati a Logs: CloudWatch

   ```
   {
       "enableNetworkPolicy": "true",
       "nodeAgent": {
           "enablePolicyEventLogs": "true"
       }
   }
   ```

Lo screenshot seguente mostra un esempio di tale scenario.


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true"
5 }
6 }

AWS CLI

- Esegui il seguente comando. AWS CLI Sostituisci `my-cluster` con il nome del cluster e l'ARN del ruolo IAM con il ruolo che stai utilizzando.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true"}}'
```

Componenti aggiuntivi autogestiti

Helm

Se è stato installato il Amazon VPC CNI plugin for Kubernetes throughhelm, è possibile aggiornare la configurazione per scrivere i registri delle politiche di rete.

- Esegui il comando seguente per abilitare la policy di rete.

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true aws-vpc-cni --namespace
 kube-system eks/aws-vpc-cni
```

kubectl

Se è stato installato il Amazon VPC CNI plugin for Kubernetes throughkubectl, è possibile aggiornare la configurazione per scrivere i registri delle politiche di rete.

1. Apri la `DaemonSet` del `aws-node` nell'editor.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Sostituisci il valore `false` con `true` nell'argomento del comando `--enable-policy-event-logs=false` in `args:` nel container `aws-network-policy-agent` nel manifesto del daemonset `aws-node` della CNI di VPC.

```
     - args:
```

```
        - --enable-policy-event-logs=true
```

Invia i log delle policy di rete ad Amazon CloudWatch Logs

Puoi monitorare i log delle politiche di rete utilizzando servizi come Amazon CloudWatch Logs. Puoi utilizzare i seguenti metodi per inviare i log delle politiche di rete a Logs. CloudWatch

Per i cluster EKS, i log delle policy si troveranno in `/aws/eks/`*`cluster-name`*`/cluster/`, mentre per i cluster K8S autogestiti i log verranno collocati in `/aws/k8s-cluster/cluster/`.

Invio dei log delle policy di rete con il Amazon VPC CNI plugin for Kubernetes

Se si abilitano le policy di rete, ai pod viene aggiunto un secondo container `aws-node` per un agente del nodo. Questo agente del nodo può inviare i registri delle politiche di rete a Logs. CloudWatch

> **Note**
>
> Solo i log delle policy di rete vengono inviati dall'agente del nodo. Gli altri log creati dalla CNI di VPC non sono inclusi.

Prerequisiti

- Aggiungi le seguenti autorizzazioni come strofa o policy separata al ruolo IAM che stai utilizzando per la CNI di VPC.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": "*"
        }
```

```
    ]
}
```

Componenti aggiuntivi di Amazon EKS

AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui desideri configurare il componente aggiuntivo CNI di Amazon VPC.
3. Seleziona la scheda Componenti aggiuntivi.
4. Seleziona la casella nella parte superiore destra della casella del componente aggiuntivo e scegli Edit (Modifica).
5. Nella pagina Configure ***name of addon*** (Configura il nome del componente aggiuntivo):

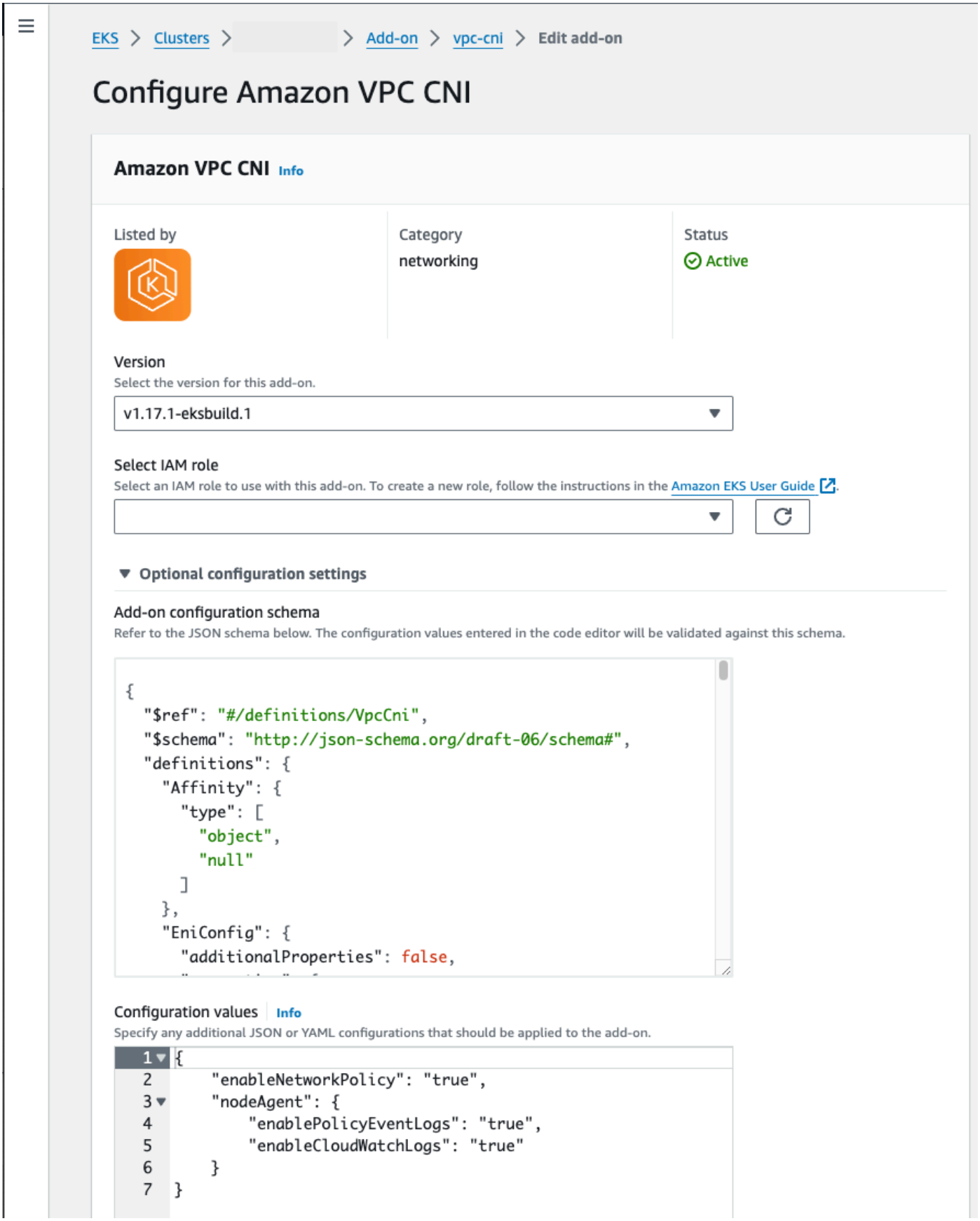
   a. Seleziona una `v1.14.0-eksbuild.3` o versione successiva nell'elenco a discesa Versione.

   b. Scegli Impostazioni di configurazione facoltative.

   c. Inserisci la chiave JSON di primo livello `"nodeAgent":` e il valore, che è un oggetto con una chiave`"enableCloudWatchLogs":` e un valore di `"true"` nei Valori di configurazione. Il testo risultante deve essere un oggetto JSON valido. L'esempio seguente mostra i criteri di rete e i registri dei criteri di rete sono abilitati e i log vengono inviati a Logs: CloudWatch

   ```
   {
       "enableNetworkPolicy": "true",
       "nodeAgent": {
           "enablePolicyEventLogs": "true",
           "enableCloudWatchLogs": "true",
       }
   }
   ```

Lo screenshot seguente mostra un esempio di tale scenario.


EKS > Clusters > > Add-on > vpc-cni > Edit add-on
Configure Amazon VPC CNI
Amazon VPC CNI Info
Listed by
Category
networking
Status
Active
Version
Select the version for this add-on.
v1.17.1-eksbuild.1
Select IAM role
Select an IAM role to use with this add-on. To create a new role, follow the instructions in the Amazon EKS User Guide.
Optional configuration settings
Add-on configuration schema
Refer to the JSON schema below. The configuration values entered in the code editor will be validated against this schema.
{
"$ref": "#/definitions/VpcCni",
"$schema": "http://json-schema.org/draft-06/schema#",
"definitions": {
"Affinity": {
"type": [
"object",
"null"
]
},
"EniConfig": {
"additionalProperties": false,
Configuration values Info
Specify any additional JSON or YAML configurations that should be applied to the add-on.
1 {
2 "enableNetworkPolicy": "true",
3 "nodeAgent": {
4 "enablePolicyEventLogs": "true",
5 "enableCloudWatchLogs": "true"
6 }
7 }

AWS CLI

- Esegui il seguente comando. AWS CLI Sostituisci `my-cluster` con il nome del cluster e l'ARN del ruolo IAM con il ruolo che stai utilizzando.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version v1.14.0-eksbuild.3 \
    --service-account-role-arn arn:aws:iam::123456789012:role/
AmazonEKSVPCCNIRole \
    --resolve-conflicts PRESERVE --configuration-values '{"nodeAgent":
 {"enablePolicyEventLogs": "true", "enableCloudWatchLogs": "true"}}'
```

Componenti aggiuntivi autogestiti

Helm

Se è stato installato il Amazon VPC CNI plugin for Kubernetes throughhelm, è possibile aggiornare la configurazione per inviare i registri delle politiche di rete a CloudWatch Logs.

- Esegui il comando seguente per abilitare i registri delle politiche di rete e inviarli a Logs. CloudWatch

```
helm upgrade --set nodeAgent.enablePolicyEventLogs=true --set
 nodeAgent.enableCloudWatchLogs=true aws-vpc-cni --namespace kube-system eks/
aws-vpc-cni
```

kubectl

1. Apri la `DaemonSet` del `aws-node` nell'editor.

```
kubectl edit daemonset -n kube-system aws-node
```

2. Sostituisci il `false` with `true` in due argomenti di comando `--enable-policy-event-logs=false` e `--enable-cloudwatch-logs=false` nel `aws-network-policy-agent` contenitore `args:` nel manifesto del daemonset CNI `aws-node` VPC.

```
    - args:
       - --enable-policy-event-logs=true
```

```
        - --enable-cloudwatch-logs=true
```

Invio dei log delle policy di rete con un daemonset Fluent Bit

Se stai utilizzando Fluent Bit in un daemonset per inviare i log dai nodi, è possibile aggiungere una configurazione che includa i log delle policy di rete provenienti dalle policy di rete. È possibile utilizzare la configurazione di esempio seguente:

```
    [INPUT]
        Name              tail
        Tag               eksnp.*
        Path              /var/log/aws-routed-eni/network-policy-agent*.log
        Parser            json
        DB                /var/log/aws-routed-eni/flb_npagent.db
        Mem_Buf_Limit     5MB
        Skip_Long_Lines   On
        Refresh_Interval  10
```

SDK eBPF incluso

Il Amazon VPC CNI plugin for Kubernetes installa la raccolta di strumenti dell'SDK eBPF sui nodi. È possibile utilizzare gli strumenti dell'SDK eBPF per identificare i problemi relativi alle policy di rete. Ad esempio, il comando seguente elenca i programmi in esecuzione sul nodo.

```
sudo /opt/cni/bin/aws-eks-na-cli ebpf progs
```

Per eseguire questo comando, è possibile utilizzare qualsiasi metodo di connessione al nodo.

Policy di rete Kubernetes

Per implementare le policy di rete Kubernetes, è necessario creare oggetti `NetworkPolicy` di Kubernetes e implementarli nel cluster. Gli oggetti `NetworkPolicy` sono riconducibili a uno spazio dei nomi. Le policy vengono implementate per consentire o negare il traffico tra Pods in base a selettori di etichette, spazi dei nomi e intervalli di indirizzi IP. Per ulteriori informazioni sulla creazione di oggetti `NetworkPolicy`, consulta la pagina Network Policies nella documentazione di Kubernetes.

L'applicazione di oggetti Kubernetes `NetworkPolicy` viene implementata con l'Extended Berkeley Packet Filter (eBPF). Relativo alle implementazioni basate su `iptables`, offre caratteristiche

di latenza e prestazioni inferiori, tra cui un utilizzo ridotto della CPU e l'evitamento delle ricerche sequenziali. Inoltre, le sonde eBPF forniscono l'accesso a dati contestuali che aiutano a eseguire il debug di problemi complessi a livello di kernel e a migliorare l'osservabilità. Amazon EKS supporta un esportatore eBPF basato su dati che sfrutta le sonde per registrare i risultati delle policy su ogni nodo ed esportare i dati in raccoglitori di log esterni per facilitare il debug. Per ulteriori informazioni, consulta la documentazione relativa ad eBPF.

Rete personalizzata per i pod

Per impostazione predefinita, Amazon VPC CNI plugin for Kubernetes crea interfacce di rete elastiche secondarie (interfacce di rete) per il nodo Amazon EC2 nella stessa sottorete dell'interfaccia di rete primaria del nodo. Inoltre, associa all'interfaccia di rete secondaria gli stessi gruppi di sicurezza associati a quella primaria. È possibile far sì che il plugin crei interfacce di rete secondarie in una sottorete diversa o associ gruppi di sicurezza diversi alle interfacce di rete secondarie, o entrambe le cose, in base a uno dei motivi riportati di seguito:

- È disponibile un numero limitato di indirizzi `IPv4` nella sottorete in cui si trova l'interfaccia di rete primaria. Ciò potrebbe limitare il numero di Pods che è possibile creare nella sottorete. Utilizzando una sottorete diversa per le interfacce di rete secondarie, è possibile aumentare il numero di indirizzi `IPv4` disponibili per i Pods.
- Per motivi di sicurezza, i Pods dovrebbero usare sottoreti o gruppi di sicurezza diversi rispetto all'interfaccia di rete primaria del nodo.
- I nodi sono configurati in sottoreti pubbliche e i Pods possono essere collocati in sottoreti private. La tabella di instradamento associata a una sottorete pubblica include un percorso a un gateway Internet. La tabella di instradamento associata a una sottorete privata non include una route a un gateway Internet.

Considerazioni

- Con una rete personalizzata abilitata, nessun indirizzo IP assegnato all'interfaccia di rete primaria viene assegnato ai Pods. Solo gli indirizzi IP delle interfacce di rete secondarie vengono assegnati ai `Pods`.
- Se il cluster utilizza la famiglia `IPv6`, non è possibile usare la rete personalizzata.
- Se si prevede di utilizzare una rete personalizzata esclusivamente per ridurre l'esaurimento dell'indirizzo `IPv4`, è invece possibile creare un cluster tramite la famiglia `IPv6`. Per ulteriori informazioni, consulta `IPv6`indirizzi per cluster Pods e services.

- I Pods implementati nelle sottoreti specificate per le interfacce di rete secondarie possono utilizzare sottoreti e gruppi di sicurezza diversi rispetto all'interfaccia di rete primaria del nodo, a patto che si trovino nello stesso VPC del nodo.

Prerequisiti

- Scopri come Amazon VPC CNI plugin for Kubernetes crea interfacce di rete secondarie e assegna gli indirizzi IP ai Pods. Per ulteriori informazioni, consulta Allocazione di ENI su GitHub.
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Ti consigliamo di completare la procedura descritta in questo argomento in una shell Bash. In alternativa, puoi apportare alcune modifiche alla tua shell per alcuni comandi di script, come i caratteri di continuazione della riga, e per il modo in cui le variabili vengono impostate e utilizzate. Inoltre, le regole di escape e di utilizzo delle virgolette per la shell (interprete di comandi) potrebbero essere diverse. Per ulteriori informazioni, vedere Uso delle virgolette con le stringhe AWS CLI nella Guida per l' AWS Command Line Interface utente.

Per questo tutorial ti consigliamo di utilizzare i *`example values`*, tranne quando è indicato di sostituirli. Puoi sostituire qualsiasi *`example value`* quando si completano i passaggi per un cluster di produzione. Ti consigliamo di completare tutti i passaggi nello stesso terminale, dato che le variabili impostate e utilizzate durante la procedura non sono presenti in terminali diversi.

I comandi di questo argomento sono formattati utilizzando le convenzioni elencate in Uso degli esempi. AWS CLI Se esegui comandi dalla riga di comando su risorse che si trovano in una Regione AWS posizione diversa da quella predefinita Regione AWS definita nel AWS CLI profilo che stai utilizzando, devi aggiungerli **`--region`** ***`region-code`*** ai comandi.

Per implementare le reti personalizzate nel cluster di produzione, passa alla Fase 2: configurazione del VPC.

Fase 1: creazione di un VPC e di un cluster di test

Come creare un cluster

Le procedure seguenti illustrano la creazione di un VPC e un cluster di test, oltre alla configurazione di una rete personalizzata per tale cluster. Ti consigliamo di non utilizzare il cluster di test per i carichi di lavoro di produzione poiché alcune funzionalità da utilizzare con il cluster di produzione non sono trattate in questo argomento. Per ulteriori informazioni, consulta Creazione di un cluster Amazon EKS.

1. Definisci alcune variabili da utilizzare nei passaggi rimanenti.

```
export cluster_name=my-custom-networking-cluster
account_id=$(aws sts get-caller-identity --query Account --output text)
```

2. Crea un VPC.

   1. Crea un VPC utilizzando un modello Amazon EKS AWS CloudFormation .

```
aws cloudformation create-stack --stack-name my-eks-custom-networking-vpc \
  --template-url https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/amazon-eks-vpc-private-subnets.yaml \
  --parameters ParameterKey=VpcBlock,ParameterValue=192.168.0.0/24 \
  ParameterKey=PrivateSubnet01Block,ParameterValue=192.168.0.64/27 \
  ParameterKey=PrivateSubnet02Block,ParameterValue=192.168.0.96/27 \
  ParameterKey=PublicSubnet01Block,ParameterValue=192.168.0.0/27 \
  ParameterKey=PublicSubnet02Block,ParameterValue=192.168.0.32/27
```

      La AWS CloudFormation creazione dello stack richiede alcuni minuti. Utilizza il comando seguente per verificare lo stato di implementazione dello stack.

```
aws cloudformation describe-stacks --stack-name my-eks-custom-networking-vpc --
query Stacks\[\].StackStatus  --output text
```

Non passare alla fase successiva finché l'output del comando non è `CREATE_COMPLETE`.

2. Definisci le variabili con i valori degli ID delle sottoreti private creati dal modello.

```
subnet_id_1=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet01'].PhysicalResourceId" --output text)
subnet_id_2=$(aws cloudformation describe-stack-resources --stack-name my-eks-
custom-networking-vpc \
    --query "StackResources[?
LogicalResourceId=='PrivateSubnet02'].PhysicalResourceId" --output text)
```

3. Definisci le variabili con le zone di disponibilità delle sottoreti recuperate nel passaggio precedente.

```
az_1=$(aws ec2 describe-subnets --subnet-ids $subnet_id_1 --query
 'Subnets[*].AvailabilityZone' --output text)
az_2=$(aws ec2 describe-subnets --subnet-ids $subnet_id_2 --query
 'Subnets[*].AvailabilityZone' --output text)
```

3. Crea un ruolo IAM del cluster.

   a. Per creare un file JSON della policy di attendibilità IAM, esegui il comando seguente.

```
cat >eks-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

   b. Crea il ruolo IAM del cluster Amazon EKS. Se necessario, inserisci il prefisso `eks-cluster-role-trust-policy.json` nel percorso del computer in cui hai eseguito la

scrittura del file nella fase precedente. Il comando associa il criterio di attendibilità creato nella fase precedente al ruolo. Per creare un ruolo IAM, è necessario assegnare l'azione `iam:CreateRole` (autorizzazione) al principale IAM che sta creando il ruolo.

```
aws iam create-role --role-name myCustomNetworkingAmazonEKSClusterRole --
assume-role-policy-document file://"eks-cluster-role-trust-policy.json"
```

c. Allega al ruolo la policy gestita da Amazon EKS, denominata `AmazonEKSClusterPolicy`. Per allegare una policy IAM a un principale IAM, è necessario assegnare al principale che sta allegando la policy una delle azioni IAM (autorizzazioni) seguenti: `iam:AttachUserPolicy` o `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSClusterPolicy --role-name myCustomNetworkingAmazonEKSClusterRole
```

4. Crea un cluster Amazon EKS e configura il dispositivo per comunicare con esso.

a. Crea un cluster.

```
aws eks create-cluster --name my-custom-networking-cluster \
   --role-arn arn:aws:iam::$account_id:role/
myCustomNetworkingAmazonEKSClusterRole \
   --resources-vpc-config subnetIds=$subnet_id_1","$subnet_id_2
```

> **Note**
>
> Potresti ricevere un messaggio di errore indicante che una delle zone di disponibilità nella richiesta non dispone di capacità sufficiente per creare un cluster Amazon EKS. In questo caso, l'output di errore contiene le zone di disponibilità in grado di supportare un nuovo cluster. Riprova a creare il cluster con almeno due sottoreti che si trovano nelle zone di disponibilità supportate per il tuo account. Per ulteriori informazioni, consulta Capacità insufficiente.

b. La creazione del cluster richiede diversi minuti. Utilizza il comando seguente per verificare lo stato di implementazione del cluster.

```
aws eks describe-cluster --name my-custom-networking-cluster --query
 cluster.status
```

Non passare alla fase successiva finché l'output del comando non è `"ACTIVE"`.

c. Configura `kubectl` per comunicare con il cluster.

```
aws eks update-kubeconfig --name my-custom-networking-cluster
```

Fase 2: configurazione del VPC

Questo tutorial richiede il VPC creato nella Fase 1: creazione di un VPC e di un cluster di test. Per un cluster di produzione, modificare i passaggi in base al VPC, sostituendo tutti i *example values* con i propri.

1. Verifica che la versione del Amazon VPC CNI plugin for Kubernetes attualmente installata sia la più recente. Per determinare la versione più recente per il componente aggiuntivo del tipo Amazon EKS e aggiornarla, consulta la pagina Aggiornamento di un componente aggiuntivo. Per determinare la versione più recente per il componente aggiuntivo del tipo autogestito e aggiornarla, consulta la pagina Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.
2. Recupera l'ID del VPC in cui si trova il cluster e memorizzarlo in una variabile per l'utilizzo nei passaggi successivi. Per un cluster di produzione, sostituisci *my-custom-networking-cluster* con il nome del cluster.

```
vpc_id=$(aws eks describe-cluster --name my-custom-networking-cluster --query
 "cluster.resourcesVpcConfig.vpcId" --output text)
```

3. Associa un blocco di instradamento interdominio senza classi (CIDR) aggiuntivo al VPC del cluster. Il blocco CIDR non può sovrapporsi a blocchi CIDR esistenti già associati.

   1. Visualizza i blocchi CIDR attualmente associati al tuo VPC.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id \
    --query 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

   Di seguito viene riportato un output di esempio:

```
---------------------------------
|          DescribeVpcs         |
```

```
+-----------------+-------------+
|    CIDRBlock    |    State    |
+-----------------+-------------+
|  192.168.0.0/24 |  associated |
+-----------------+-------------+
```

2. Associa un blocco CIDR aggiuntivo al VPC. Per ulteriori informazioni, consulta Associazione di blocchi CIDR IPv4 aggiuntivi al VPC nella Guida per l'utente di Amazon VPC.

```
aws ec2 associate-vpc-cidr-block --vpc-id $vpc_id --cidr-block 192.168.1.0/24
```

3. Verifica che il nuovo blocco sia associato.

```
aws ec2 describe-vpcs --vpc-ids $vpc_id --query
 'Vpcs[*].CidrBlockAssociationSet[*].{CIDRBlock: CidrBlock, State:
 CidrBlockState.State}' --out table
```

   Di seguito viene riportato un output di esempio:

```
---------------------------------
|          DescribeVpcs         |
+-----------------+-------------+
|    CIDRBlock    |    State    |
+-----------------+-------------+
|  192.168.0.0/24 |  associated |
|  192.168.1.0/24 |  associated |
+-----------------+-------------+
```

   Non andare al passaggio successivo finché il nuovo `State` del blocco CIDR non è `associated`.

4. Crea tutte le sottoreti che vuoi utilizzare in ogni zona di disponibilità in cui si trovano le sottoreti esistenti. Specifica un blocco CIDR all'interno del blocco CIDR associato al VPC in una fase precedente.

   1. Crea nuove sottoreti. È necessario creare le sottoreti in un blocco CIDR VPC diverso rispetto alle sottoreti esistenti, ma all'interno delle stesse zone di disponibilità. In questo esempio si crea una sottorete nel nuovo blocco CIDR in ogni zona di disponibilità in cui esistono le sottoreti private attuali. Gli ID delle sottoreti create vengono memorizzati in variabili da utilizzare nei passaggi successivi. I valori `Name` corrispondono a quelli assegnati alle sottoreti

create utilizzando il modello Amazon EKS VPC in un passaggio precedente. I nomi non sono obbligatori. È possibile utilizzare nomi diversi.

```
new_subnet_id_1=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone
 $az_1 --cidr-block 192.168.1.0/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-
custom-networking-vpc-PrivateSubnet01},{Key=kubernetes.io/role/internal-
elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
new_subnet_id_2=$(aws ec2 create-subnet --vpc-id $vpc_id --availability-zone
 $az_2 --cidr-block 192.168.1.32/27 \
    --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=my-eks-
custom-networking-vpc-PrivateSubnet02},{Key=kubernetes.io/role/internal-
elb,Value=1}]' \
    --query Subnet.SubnetId --output text)
```

> **Important**
>
> Per impostazione predefinita, le nuove sottoreti sono implicitamente associate alla tabella di instradamento principale del VPC. Questa tabella di instradamento consente la comunicazione tra tutte le risorse implementate nel VPC, ad eccezione delle risorse con indirizzi IP esterni ai blocchi CIDR associate al VPC. Per modificare questo comportamento, associa la tabella di instradamento alle sottoreti. Per ulteriori informazioni, consulta Tabelle di instradamento per sottoreti nella Guida per l'utente di Amazon VPC.

2. Per visualizzare le sottoreti attuali nel tuo VPC.

```
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" \
    --query 'Subnets[*].{SubnetId: SubnetId,AvailabilityZone:
 AvailabilityZone,CidrBlock: CidrBlock}' \
    --output table
```

Di seguito viene riportato un output di esempio:

```
-------------------------------------------------------------------------
|                           DescribeSubnets                             |
+------------------+--------------------+---------------------------+
| AvailabilityZone |     CidrBlock      |         SubnetId          |
+------------------+--------------------+---------------------------+
```

```
|  us-west-2d      |  192.168.0.0/27    |     subnet-example1        |
|  us-west-2a      |  192.168.0.32/27   |     subnet-example2        |
|  us-west-2a      |  192.168.0.64/27   |     subnet-example3        |
|  us-west-2d      |  192.168.0.96/27   |     subnet-example4        |
|  us-west-2a      |  192.168.1.0/27    |     subnet-example5        |
|  us-west-2d      |  192.168.1.32/27   |     subnet-example6        |
+------------------+--------------------+----------------------------+
```

Come puoi vedere, le sottoreti create nel blocco CIDR `192.168.1.0` si trovano nelle stesse zone di disponibilità di quelle create nel blocco CIDR `192.168.0.0`.

Fase 3: configurazione di risorse Kubernetes

Configurazione di risorse Kubernetes

1. Imposta la variabile di ambiente `AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG` su `true` nel `aws-node DaemonSet`.

```
kubectl set env daemonset aws-node -n kube-system
 AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG=true
```

2. Recupera l'ID del gruppo di sicurezza del cluster e memorizzalo in una variabile da utilizzare nel passaggio successivo. Amazon EKS crea automaticamente questo gruppo di sicurezza durante la creazione del cluster.

```
cluster_security_group_id=$(aws eks describe-cluster --name $cluster_name --query
 cluster.resourcesVpcConfig.clusterSecurityGroupId --output text)
```

3. Crea una risorsa personalizzata `ENIConfig` per ogni sottorete in cui vuoi implementare i Pods.

   a. Crea un file univoco per ogni configurazione di interfaccia di rete.

   I comandi seguenti creano file `ENIConfig` separati per le due sottoreti create in una fase precedente. Il valore di `name` deve essere univoco. Il nome corrisponde alla zona di disponibilità in cui si trova la sottorete. Il gruppo di sicurezza del cluster viene assegnato a `ENIConfig`.

```
cat >$az_1.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
```

```
metadata:
  name: $az_1
spec:
  securityGroups:
    - $cluster_security_group_id
  subnet: $new_subnet_id_1
EOF
```

```
cat >$az_2.yaml <<EOF
apiVersion: crd.k8s.amazonaws.com/v1alpha1
kind: ENIConfig
metadata:
  name: $az_2
spec:
  securityGroups:
    - $cluster_security_group_id
  subnet: $new_subnet_id_2
EOF
```

Per un cluster di produzione, puoi apportare le seguenti modifiche ai comandi precedenti:

- Sostituisci *$cluster_security_group_id* con l'ID di un gruppo di sicurezza esistente da utilizzare per ciascun `ENIConfig`.
- Quando possibile, assegna a `ENIConfigs` lo stesso nome della zona di disponibilità che utilizzerai per `ENIConfig`. Per una serie di motivi, tuttavia, potrebbe essere necessario utilizzare nomi diversi per `ENIConfigs` e le zone di disponibilità. Ad esempio, se nella stessa zona di disponibilità sono presenti più di due sottoreti e si desidera utilizzarle entrambe con reti personalizzate, saranno necessari più `ENIConfigs` per la stessa zona di disponibilità. Dal momento che ogni `ENIConfig` richiede un nome univoco, non puoi assegnare lo stesso nome della zona di disponibilità a più `ENIConfigs`.

  Se i nomi di `ENIConfig` non sono tutti uguali a quelli delle zone di disponibilità, sostituisci *$az_1* e *$az_2* con i nomi scelti nei comandi precedenti e prendi nota dei nodi con `ENIConfig` come descritto più avanti in questo tutorial.

> **Note**
>
> Nel caso in cui non si specifichi un gruppo di sicurezza valido da utilizzare con un cluster di produzione:

- se si utilizza la versione `1.8.0` o successiva del Amazon VPC CNI plugin for Kubernetes, vengono impiegati i gruppi di sicurezza associati all'interfaccia di rete elastica primaria del nodo.
- se si utilizza una versione del Amazon VPC CNI plugin for Kubernetes precedente alla `1.8.0`, il gruppo di sicurezza di default per il VPC viene assegnato alle interfacce di rete secondarie.

Important

- `AWS_VPC_K8S_CNI_EXTERNALSNAT=false` è un'impostazione predefinita per la configurazione del plugin CNI di Amazon VPC per Kubernetes. Per impostazione predefinita, il traffico destinato agli indirizzi IP che non si trovano all'interno di uno dei blocchi CIDR associati al VPC utilizza i gruppi di sicurezza e le sottoreti dell'interfaccia di rete primaria del nodo. Le sottoreti e i gruppi di sicurezza definiti in `ENIConfigs` al fine di creare interfacce di rete secondarie non vengono utilizzati per questo traffico. Per ulteriori informazioni su questa impostazione, consulta SNAT per Pods.
- Se si utilizzano gruppi di sicurezza anche per i Pods, il gruppo di sicurezza specificato in una `SecurityGroupPolicy` viene utilizzato al posto di quello specificato in `ENIConfigs`. Per ulteriori informazioni, consulta Gruppi di sicurezza per Pods.

b. Applica ogni file delle risorse personalizzato creato in precedenza al cluster con i comandi seguenti.

```
kubectl apply -f $az_1.yaml
kubectl apply -f $az_2.yaml
```

4. Verifica che `ENIConfigs` sia stato creato.

```
kubectl get ENIConfigs
```

Di seguito viene riportato un output di esempio:

```
NAME         AGE
```

```
us-west-2a   117s
us-west-2d   105s
```

5. Se stai abilitando la rete personalizzata su un cluster di produzione e hai assegnato a `ENIConfigs` un nome diverso da quello dalla zona di disponibilità per cui lo stai utilizzando, passa alla fase successiva per implementare i nodi Amazon EC2.

   Consenti a Kubernetes di applicare automaticamente `ENIConfig` per una zona di disponibilità a tutti i nuovi nodi Amazon EC2 creati nel cluster.

   1. Per il cluster di test in questo tutorial, passa alla fase successiva.

      Per il cluster di produzione, verifica la presenza di un'annotation con il codice `k8s.amazonaws.com/eniConfig` per la variabile di ambiente `ENI_CONFIG_ANNOTATION_DEF` nelle specifiche del container per `aws-node DaemonSet`.

      ```
      kubectl describe daemonset aws-node -n kube-system | grep
       ENI_CONFIG_ANNOTATION_DEF
      ```

      Se viene restituito un output, l'annotazione è presente. Se non viene restituito alcun output, la variabile non viene impostata. Per un cluster di produzione, puoi scegliere di utilizzare la presente impostazione o quella riportata nel passaggio successivo. L'utilizzo di questa impostazione sovrascrive quella del passaggio successivo. In questo tutorial viene utilizzata l'impostazione riportata nel passaggio successivo.

   2. Aggiorna il tuo `aws-node DaemonSet` per applicare automaticamente `ENIConfig` per una zona di disponibilità a tutti i nuovi nodi Amazon EC2 creati nel cluster.

      ```
      kubectl set env daemonset aws-node -n kube-system
       ENI_CONFIG_LABEL_DEF=topology.kubernetes.io/zone
      ```

Fase 4: implementazione di nodi Amazon EC2

Per implementare i nodi Amazon EC2

1. Creare un ruolo IAM del nodo.

   a. Per creare un file JSON della policy di attendibilità IAM, esegui il comando seguente.

      ```
      cat >node-role-trust-relationship.json <<EOF
      ```

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

b. Eseguire questo comando per impostare una variabile per il nome del ruolo. Puoi sostituire *myCustomNetworkingAmazonEKSNodeRole* con un nome a tua scelta.

```
export node_role_name=myCustomNetworkingAmazonEKSNodeRole
```

c. Crea il ruolo IAM e memorizza il nome della risorsa Amazon (ARN) restituito in una variabile da utilizzare in una fase successiva.

```
node_role_arn=$(aws iam create-role --role-name $node_role_name --assume-role-
policy-document file://"node-role-trust-relationship.json" \
    --query Role.Arn --output text)
```

d. Allega al ruolo IAM le tre policy gestite IAM richieste.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name $node_role_name
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name $node_role_name
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
    --role-name $node_role_name
```

> **Important**
>
> Per semplicità, in questo tutorial la policy AmazonEKS_CNI_Policy è allegata al ruolo IAM del nodo. Tuttavia, in un cluster di produzione, si consiglia di allegare la

policy a un ruolo IAM separato da utilizzare solo con Amazon VPC CNI plugin for Kubernetes. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

2. Creare uno dei seguenti tipi di gruppi di nodi. Per determinare il tipo di istanza da implementare, consulta Scelta di un tipo di istanza Amazon EC2. Per questo tutorial, utilizza l'opzione Gestito, senza un modello di avvio o con un modello di avvio senza un ID AMI specificato. Se prevedi di utilizzare il gruppo di nodi per i carichi di lavoro di produzione, ti consigliamo di familiarizzare con tutte le opzioni del gruppo di nodi gestito e autogestito prima di implementare il gruppo di nodi.

   - Gestito: implementare il gruppo di nodi utilizzando una delle opzioni seguenti:
     - Senza un modello di avvio o con un modello di avvio senza un ID AMI specificato: esegui il comando seguente. Per questo tutorial, utilizza i *example values*. Per un gruppo di nodi di produzione, sostituisci tutti i *example values* con i tuoi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura.

       ```
       aws eks create-nodegroup --cluster-name $cluster_name --nodegroup-name my-
       nodegroup \
           --subnets $subnet_id_1 $subnet_id_2 --instance-types t3.medium --node-role
        $node_role_arn
       ```

     - Con un modello di avvio con un ID AMI specificato
       1. Determina il numero massimo di Pods consigliato da Amazon EKS per i nodi. Segui le istruzioni riportate in Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods, aggiungendo **`--cni-custom-networking-enabled`** alla fase 3 di questo argomento. Annota l'output restituito per l'utilizzo in un passaggio successivo.
       2. Nel modello di avvio, specifica un ID AMI ottimizzato per Amazon EKS o un'AMI personalizzata sviluppata con l'AMI ottimizzato per Amazon EKS, quindi implementa il gruppo di nodi utilizzando un modello di avvio e fornisci i seguenti dati utente nel modello di avvio. Questi dati utente passano argomenti nel file `bootstrap.sh`. Per ulteriori informazioni sul file bootstrap, consulta bootstrap.sh su GitHub. Puoi sostituire *20* con il valore del passaggio precedente (consigliato) o con un tuo valore desiderato.

          ```
          /etc/eks/bootstrap.sh my-cluster --use-max-pods false --kubelet-extra-args
           '--max-pods=20'
          ```

Se si è creata un'AMI personalizzata che non è stata sviluppata con l'AMI ottimizzata per Amazon EKS, bisogna creare autonomamente la configurazione.

- Autogestito

  1. Determina il numero massimo di Pods consigliato da Amazon EKS per i nodi. Segui le istruzioni riportate in Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods, aggiungendo **`--cni-custom-networking-enabled`** alla fase 3 di questo argomento. Annota l'output restituito per l'utilizzo in un passaggio successivo.
  2. Implementa il gruppo di nodi utilizzando le istruzioni in Avvio di nodi Amazon Linux autogestiti. Specificate il testo seguente per il BootstrapArgumentsparametro. Puoi sostituire *`20`* con il valore del passaggio precedente (consigliato) o con un tuo valore desiderato.

     ```
     --use-max-pods false --kubelet-extra-args '--max-pods=20'
     ```

> **Note**
>
> Se vuoi che i nodi presenti in un cluster di produzione supportino un numero significativamente più elevato di Pods, esegui nuovamente lo script in Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods. Aggiungi inoltre l'opzione **`--cni-prefix-delegation-enabled`** al comando. Ad esempio, per un tipo di istanza `m5.large` viene restituito *`110`*. Per istruzioni su come abilitare questa funzionalità, consulta Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2. Puoi utilizzare questa funzionalità con una rete personalizzata.

La creazione di gruppi di nodi richiede diversi minuti. Puoi verificare lo stato della creazione di un gruppo di nodi gestiti con il comando seguente.

```
aws eks describe-nodegroup --cluster-name $cluster_name --nodegroup-name my-
nodegroup --query nodegroup.status --output text
```

Non andare al passaggio successivo finché l'output restituito non è `ACTIVE`.

3. Ai fini di questo tutorial, puoi ignorare questo passaggio.

   Per un cluster di produzione, se hai assegnato a `ENIConfigs` un nome diverso da quello della zona di disponibilità in cui lo stai utilizzando, devi annotare i nodi con il nome `ENIConfig`

da utilizzare con il nodo. Questo passaggio non è necessario se disponi di una sola sottorete in ogni zona di disponibilità e hai assegnato a `ENIConfigs` gli stessi nomi delle zone di disponibilità. Questo perché il Amazon VPC CNI plugin for Kubernetes associa automaticamente il `ENIConfig` corretto al nodo se questa opzione è stata abilitata in una fase precedente.

a. Ottieni l'elenco dei nodi nel cluster.

```
kubectl get nodes
```

Di seguito viene riportato un output di esempio:

```
NAME                                          STATUS   ROLES    AGE     VERSION
ip-192-168-0-126.us-west-2.compute.internal   Ready    <none>   8m49s
 v1.22.9-eks-810597c
ip-192-168-0-92.us-west-2.compute.internal    Ready    <none>   8m34s
 v1.22.9-eks-810597c
```

b. Determina in quale zona di disponibilità si trova ogni nodo. Esegui il comando seguente per ogni nodo restituito nel passaggio precedente.

```
aws ec2 describe-instances --filters Name=network-interface.private-dns-
name,Values=ip-192-168-0-126.us-west-2.compute.internal \
--query 'Reservations[].Instances[].{AvailabilityZone:
 Placement.AvailabilityZone, SubnetId: SubnetId}'
```

Di seguito viene riportato un output di esempio:

```
[
    {
        "AvailabilityZone": "us-west-2d",
        "SubnetId": "subnet-Example5"
    }
]
```

c. Annota ogni nodo con la risorsa `ENIConfig` creata per l'ID della sottorete e la zona di disponibilità. Puoi annotare solo un nodo con un `ENIConfig`, ma più nodi con lo stesso `ENIConfig`. Sostituisci i *`example values`* con i valori in tuo possesso.

```
kubectl annotate node ip-192-168-0-126.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName1
```

```
kubectl annotate node ip-192-168-0-92.us-west-2.compute.internal
 k8s.amazonaws.com/eniConfig=EniConfigName2
```

4. Se in un cluster di produzione erano presenti nodi con Pods in esecuzione prima di passare all'utilizzo della funzionalità di rete personalizzata, completa le attività seguenti:

   a. Assicurati di avere dei nodi disponibili abilitati per la funzionalità di rete personalizzata.

   b. Cordonare ed espellere i nodi per interrompere normalmente i Pods. Per ulteriori informazioni, consulta Espulsione di un nodo in modo sicuro nella documentazione di Kubernetes.

   c. Termina i nodi. Se i nodi si trovano in un gruppo di nodi gestito esistente, puoi eliminare il gruppo di nodi. Copia il comando seguente sul tuo dispositivo. Apportare le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato:

      - Sostituisci *my-cluster* con il nome del tuo cluster.
      - Sostituisci *my-nodegroup* con un nome per il gruppo di nodi.

```
aws eks delete-nodegroup --cluster-name my-cluster --nodegroup-name my-
nodegroup
```

   Solo i nuovi nodi registrati con l'etichetta `k8s.amazonaws.com/eniConfig` useranno la nuova funzionalità di rete personalizzata.

5. Verifica che ai Pods venga assegnato un indirizzo IP da un blocco CIDR associato a una delle sottoreti create in una fase precedente.

```
kubectl get pods -A -o wide
```

   Di seguito viene riportato un output di esempio:

```
NAMESPACE     NAME                          READY   STATUS    RESTARTS   AGE     IP
           NODE                                                NOMINATED NODE   READINESS
 GATES
kube-system   aws-node-2rkn4                1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
```

```
kube-system   aws-node-k96wp             1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
kube-system   coredns-657694c6f4-smcgr   1/1     Running   0          56m
 192.168.1.23    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   coredns-657694c6f4-stwv9   1/1     Running   0          56m
 192.168.1.28    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-jgshq           1/1     Running   0          7m19s
 192.168.0.92    ip-192-168-0-92.us-west-2.compute.internal    <none>
 <none>
kube-system   kube-proxy-wx9vk           1/1     Running   0          7m15s
 192.168.0.126   ip-192-168-0-126.us-west-2.compute.internal   <none>
 <none>
```

Ai Pods `coredns` vengono assegnati indirizzi IP dal blocco CIDR `192.168.1.0` aggiunto al VPC. Senza una rete personalizzata, ai pod sarebbero stati assegnati indirizzi provenienti dall'unico blocco CIDR originariamente associato al VPC, ossia il blocco CIDR `192.168.0.0`.

Se una `spec` del Pod's contiene il valore `hostNetwork=true`, viene assegnato l'indirizzo IP principale del nodo, non un indirizzo proveniente dalle sottoreti aggiunte. Per impostazione predefinita, questo valore è impostato su `false`. Questo valore è impostato su `true` per i Pods `kube-proxy` e Amazon VPC CNI plugin for Kubernetes (`aws-node`) in esecuzione nel cluster. Per questo motivo al `kube-proxy` e ai Pods `aws-node` del plugin non vengono assegnati indirizzi `192.168.1.`*x* nell'output precedente. Per ulteriori informazioni su un'Pod's`hostNetwork`impostazione, consulta PodSpec v1 core nel riferimento all'KubernetesAPI.

Fase 5: eliminazione delle risorse del tutorial

Ti consigliamo di eliminare le risorse create dopo aver completato il tutorial. In seguito, potrai modificare i passaggi per abilitare la rete personalizzata di un cluster di produzione.

Per eliminare le risorse del tutorial

1. Se il gruppo di nodi creato era solo a scopo di test, eliminalo.

```
aws eks delete-nodegroup --cluster-name $cluster_name --nodegroup-name my-nodegroup
```

Anche dopo che l' AWS CLI output indica che il cluster è stato eliminato, il processo di eliminazione potrebbe non essere effettivamente completo. Questo processo di eliminazione richiede alcuni minuti. Verifica che il processo sia completo con il comando seguente.

```
aws eks describe-nodegroup --cluster-name $cluster_name --nodegroup-name my-
nodegroup --query nodegroup.status --output text
```

Non continuare finché l'output restituito non è simile al seguente.

```
An error occurred (ResourceNotFoundException) when calling the DescribeNodegroup
 operation: No node group found for name: my-nodegroup.
```

2. Se il gruppo di nodi creato era solo a scopo di test, elimina il ruolo IAM del nodo.

   a. Scollega le policy dal ruolo.

   ```
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
   policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
   policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly
   aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSNodeRole --
   policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy
   ```

   b. Elimina il ruolo.

   ```
   aws iam delete-role --role-name myCustomNetworkingAmazonEKSNodeRole
   ```

3. Elimina il cluster.

   ```
   aws eks delete-cluster --name $cluster_name
   ```

   Verifica l'eliminazione del cluster con il comando seguente.

   ```
   aws eks describe-cluster --name $cluster_name --query cluster.status --output text
   ```

   Quando viene restituito un output simile al seguente, il cluster è stato eliminato correttamente.

   ```
   An error occurred (ResourceNotFoundException) when calling the DescribeCluster
    operation: No cluster found for name: my-cluster.
   ```

4. Elimina il ruolo IAM del cluster.

    a. Scollega le policy dal ruolo.

    ```
    aws iam detach-role-policy --role-name myCustomNetworkingAmazonEKSClusterRole
     --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy
    ```

    b. Elimina il ruolo.

    ```
    aws iam delete-role --role-name myCustomNetworkingAmazonEKSClusterRole
    ```

5. Elimina le sottoreti create in una fase precedente.

    ```
    aws ec2 delete-subnet --subnet-id $new_subnet_id_1
    aws ec2 delete-subnet --subnet-id $new_subnet_id_2
    ```

6. Elimina il VPC creato.

    ```
    aws cloudformation delete-stack --stack-name my-eks-custom-networking-vpc
    ```

Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2

Ogni istanza Amazon EC2 supporta un numero massimo di interfacce di rete elastiche e un numero massimo di indirizzi IP che possono essere assegnati a ogni interfaccia di rete. Ogni nodo richiede un indirizzo IP per ogni interfaccia di rete. Tutti gli altri indirizzi IP disponibili possono essere assegnati a `Pods`. Ogni `Pod` necessita di un proprio indirizzo IP. Di conseguenza, potresti avere nodi con risorse di calcolo e memoria disponibili, ma che non possono ospitare ulteriori `Pods` perché il nodo ha esaurito gli indirizzi IP a cui assegnare `Pods`.

In questo argomento, imparerai come aumentare sensibilmente il numero di indirizzi IP che i nodi possono assegnare ai `Pods` assegnando prefissi IP ai tuoi nodi, anziché assegnare singoli indirizzi IP secondari. Ogni prefisso include diversi indirizzi IP. Se non configuri il cluster per l'assegnazione del prefisso IP, il cluster deve effettuare più chiamate di interfaccia di programmazione dell’applicazione (API) di Amazon EC2 per la configurazione delle interfacce di rete e degli indirizzi IP necessari alla connettività del Pod. La frequenza di queste chiamate API, man mano che i cluster acquisiscono dimensioni maggiori, può portare a tempi di avvio di Pod e delle istanze più lunghi. Ciò si traduce in ritardi di dimensionamento per soddisfare la domanda di carichi di lavoro grandi e spigolosi e aggiunge costi generali di gestione in quanto è necessario eseguire il provisioning di cluster e VPC

aggiuntivi per soddisfare i requisiti di scalabilità. Per ulteriori informazioni, vedere Soglie Kubernetes di scalabilità su. GitHub

Considerazioni

- Ogni tipo di istanza Amazon EC2 supporta un numero massimo di Pods. Se il gruppo di nodi gestito è costituito da più tipi di istanza, a tutti i nodi del cluster viene applicato il numero minimo di Pods massimo per un'istanza del cluster.
- Per impostazione predefinita, il numero massimo di `Pods` che puoi eseguire su un nodo è pari a 110, ma puoi modificare questo numero. Se cambi il numero e disponi di un gruppo di nodi gestiti, il prossimo aggiornamento dell'AMI o del modello di avvio del gruppo di nodi comporta la creazione di nuovi nodi con il valore modificato.
- Quando passi dall'assegnazione di indirizzi IP all'assegnazione di prefissi IP, si consiglia di creare nuovi gruppi di nodi per aumentare il numero di indirizzi IP disponibili, anziché effettuare una sostituzione progressiva dei nodi esistenti. L'esecuzione di Pods su un nodo a cui sono stati assegnati sia indirizzi IP che prefissi può portare a un'incoerenza nella capacità degli indirizzi IP pubblicizzati, con un impatto sui carichi di lavoro futuri sul nodo. Per quanto riguarda le modalità di esecuzione della transizione consigliate, consulta Sostituisci tutti i nodi durante la migrazione dalla modalità IP secondario alla modalità Delega del prefisso o viceversa nella guida alle best practice di Amazon EKS.
- Solo per i cluster con nodi Linux.
  - Dopo aver configurato il componente aggiuntivo per assegnare prefissi alle interfacce di rete, non è possibile eseguire il downgrade del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes a una versione inferiore a `1.9.0` (o `1.10.1`) senza rimuovere tutti i nodi in tutti i gruppi di nodi nel cluster.
  - Se utilizzi anche gruppi di sicurezza per i Pods, con `POD_SECURITY_GROUP_ENFORCING_MODE=standard` e `AWS_VPC_K8S_CNI_EXTERNALSNAT=false`, quando i Pods comunicano con endpoint esterni al tuo VPC, vengono utilizzati i gruppi di sicurezza del nodo, anziché i gruppi di sicurezza che hai assegnato ai Pods.

    Se utilizzi anche gruppi di sicurezza per Pods, with `POD_SECURITY_GROUP_ENFORCING_MODE=strict`, quando i `Pods` comunicano con endpoint esterni al tuo VPC, vengono utilizzati i gruppi `Pod's` di sicurezza.

Prerequisiti

- Un cluster esistente. Per implementarne uno, consulta Creazione di un cluster Amazon EKS.
- Le sottoreti in cui si trovano i tuoi nodi Amazon EKS devono avere un numero sufficiente di blocchi di routing interdominio senza classi (CIDR) `/28` (per i cluster `IPv4`) o `/80` (per i cluster `IPv6`). Puoi disporre solo di nodi Linux in un cluster `IPv6`. L'uso dei prefissi IP può non riuscire se gli indirizzi IP sono sparsi nella sottorete CIDR. È preferibile quanto segue:
  - L'utilizzo di una prenotazione di sottorete CIDR in modo che, anche se alcuni indirizzi IP all'interno dell'intervallo riservato sono ancora in uso, al momento del rilascio, gli indirizzi IP non vengano riassegnati. Ciò assicura che i prefissi siano disponibili per l'assegnazione senza segmentazione.
  - Usa nuove sottoreti utilizzate specificamente per l'esecuzione di carichi di lavoro a cui sono assegnati i prefissi IP. I carichi di lavoro sia di Windows sia di Linux possono essere eseguiti nella stessa sottorete quando si assegnano prefissi IP.
- Per assegnare prefissi IP ai nodi, questi devono essere basati su Nitro. AWS Le istanze che non sono basate su Nitro continuano ad assegnare singoli indirizzi IP secondari, ma hanno un numero di gran lunga inferiore di indirizzi IP da assegnare ai Pods rispetto alle istanze Nitro-based.
- Solo per cluster con nodi Linux: se il cluster è configurato per la famiglia `IPv4`, è necessario che sia installata la versione `1.9.0` o una versione successiva del componente aggiuntivo di Amazon VPC CNI plugin for Kubernetes. Puoi controllare la tua versione attuale con il seguente comando.

  ```
  kubectl describe daemonset aws-node --namespace kube-system | grep Image | cut -d "/"
   -f 2
  ```

  Se il cluster è configurato per la famiglia `IPv6`, è necessario che sia installata la versione `1.10.1` del componente aggiuntivo. Se la versione del plugin è precedente a quelle richieste, è necessario aggiornarla. Per ulteriori informazioni, consultare le sezioni sugli aggiornamenti di Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.
- Solo per i cluster con nodi Windows
  - La versione del cluster e la versione della piattaforma relativa devono essere uguali o successive alle versioni indicate nella tabella seguente. Per aggiornare la versione del cluster, consulta Aggiornamento della versione di Kubernetes del cluster Amazon EKS. Se il cluster non è alla versione minima della piattaforma, non puoi assegnare i prefissi IP ai nodi finché Amazon EKS non ha aggiornato la versione della piattaforma.

Versione di Kubernetes	Versione della piattaforma
1.27	eks.3
1.26	eks.4
1.25	eks.5

Puoi verificare la versione della piattaforma e quella di Kubernetes attuali sostituendo *my-cluster* nel seguente comando con il nome del tuo cluster e quindi eseguendo il comando modificato: **aws eks describe-cluster --name *my-cluster* --query 'cluster.{"Kubernetes Version": version, "Platform Version": platformVersion}'**.

- Supporto Windows abilitato per il cluster. Per ulteriori informazioni, consulta Abilitazione del supporto di Windows per il cluster Amazon EKS.

Per aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2

1. Configura il cluster per assegnare i prefissi degli indirizzi IP ai nodi. Completa la procedura nella scheda corrispondente al sistema operativo del nodo.

   Linux

   1. Abilitare il parametro per assegnare prefissi alle interfacce di rete per il `DaemonSet` CNI di Amazon VPC. Quando implementi un cluster versione `1.21` o successiva, una versione `1.10.1` o successiva del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes viene implementata di conseguenza. Se hai creato il cluster con la famiglia `IPv6`, questa impostazione era su `true` di default. Se hai creato il cluster con la famiglia `IPv4`, questa impostazione era su `false` di default.

      ```
      kubectl set env daemonset aws-node -n kube-system
       ENABLE_PREFIX_DELEGATION=true
      ```

> **Important**
>
> Anche se la sottorete dispone di indirizzi IP disponibili, se non dispone di blocchi `/28` contigui disponibili, verrà visualizzato il seguente errore nei log del Amazon VPC CNI plugin for Kubernetes.
>
> ```
> InsufficientCidrBlocks: The specified subnet does not have enough free
>  cidr blocks to satisfy the request
> ```
>
> Ciò può verificarsi a causa della frammentazione degli indirizzi IP secondari esistenti distribuiti in una sottorete. Per risolvere questo errore, creare una nuova sottorete e avviare i Pods lì oppure utilizzare una prenotazione CIDR della sottorete Amazon EC2 per riservare spazio all'interno di una sottorete da utilizzare con l'assegnazione del prefisso. Per ulteriori informazioni, consultare la sezione relativa a Prenotazioni della CIDR per la sottorete nella Guida per l'utente di Amazon VPC.

2. Se si intende implementare un gruppo di nodi gestito senza un modello di avvio o con un modello di avvio in cui non è specificato un ID AMI e si sta utilizzando una versione del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes corrispondente o successiva alle versioni riportate nei prerequisiti, andare al passaggio successivo. I gruppi di nodi gestiti calcolano automaticamente il numero massimo di Pods.

   Se si sta implementando un gruppo di nodi autogestito o un gruppo di nodi gestito con un modello di avvio in cui è specificato un ID AMI, è necessario determinare il numero massimo di Pods consigliati da Amazon EKS per i nodi. Seguire le istruzioni riportate in Per ogni tipo di istanza Amazon EC2, Amazon EKS consiglia un numero massimo di Pods, aggiungendo **`--cni-prefix-delegation-enabled`** al passaggio 3. Annotare l'output restituito per l'utilizzo in un passaggio successivo.

   > **Important**
   >
   > I gruppi di nodi gestiti applicano un numero massimo sul valore di `maxPods`. Per le istanze con meno di 30 vCPUs il numero massimo è 110 e per tutte le altre istanze il numero massimo è 250. Questo numero massimo viene applicato indipendentemente dal fatto che la delega del prefisso sia abilitata o meno.

3. Se si usa un cluster `1.21` o successivo configurato per `IPv6`, andare al passaggio successivo.

   É possibile specificare i parametri una delle seguenti opzioni. Per determinare quale opzione è più adatta e quale valore fornire per essa, consulta `WARM_PREFIX_TARGET`, `WARM_IP_TARGET` e `MINIMUM_IP_TARGET` su GitHub.

   Puoi sostituire i *`example values`* con un valore maggiore di zero.

   - `WARM_PREFIX_TARGET`

     ```
     kubectl set env ds aws-node -n kube-system WARM_PREFIX_TARGET=1
     ```

   - `WARM_IP_TARGET` o `MINIMUM_IP_TARGET` – Se uno dei due valori è impostato, sovrascrive qualsiasi valore impostato per `WARM_PREFIX_TARGET`.

     ```
     kubectl set env ds aws-node -n kube-system WARM_IP_TARGET=5
     ```

     ```
     kubectl set env ds aws-node -n kube-system MINIMUM_IP_TARGET=2
     ```

4. Creare uno dei seguenti tipi di gruppi di nodi con almeno un tipo di istanza Amazon EC2 Nitro Amazon Linux 2. Per un elenco dei tipi di istanze Nitro, consulta Instances built on the Nitro System nella Amazon EC2 User Guide. Questa funzionalità non è supportata su Windows. Per le opzioni che includono *`110`*, sostituirlo con il valore del passaggio 3 (consigliato) o con il proprio valore.

   - Autogestito – Implementare il gruppo di nodi utilizzando le istruzioni contenute in Avvio di nodi Amazon Linux autogestiti. Specificare il testo seguente per il parametro. BootstrapArguments

     ```
     --use-max-pods false --kubelet-extra-args '--max-pods=110'
     ```

     Se stai utilizzando `eksctl` per creare il gruppo di nodi, puoi usare il seguente comando.

     ```
     eksctl create nodegroup --cluster my-cluster --managed=false --max-pods-per-node 110
     ```

   - Gestito – Implementare il gruppo di nodi utilizzando una delle opzioni seguenti:

     - Senza un modello di avvio o con un modello di avvio senza un ID AMI specificato – Completare la procedura in Creazione di un gruppo di nodi gestiti. I gruppi di nodi

gestiti calcolano automaticamente il valore massimo di `max-pods` consigliato da Amazon EKS.

- Con un modello di avvio con un ID AMI specificato – nel modello di lancio, specificare un ID AMI ottimizzato per Amazon EKS o un'AMI personalizzata sviluppata con l'AMI ottimizzato Amazon EKS, quindi implementare il gruppo di nodi utilizzando un modello di avvio e fornire i seguenti dati utente nel modello di avvio. Questi dati utente passano argomenti nel file `bootstrap.sh`. Per ulteriori informazioni sul file bootstrap, consulta bootstrap.sh su GitHub.

```
/etc/eks/bootstrap.sh my-cluster \
  --use-max-pods false \
  --kubelet-extra-args '--max-pods=110'
```

Se stai utilizzando `eksctl` per creare il gruppo di nodi, puoi usare il seguente comando.

```
eksctl create nodegroup --cluster my-cluster --max-pods-per-node 110
```

Se si è creata un'AMI personalizzata che non è stata sviluppata con l'AMI ottimizzata per Amazon EKS, bisogna creare autonomamente la configurazione.

> **Note**
>
> Se si desidera assegnare anche indirizzi IP ai Pods di una sottorete diversa da quella dell'istanza, è necessario abilitare la funzionalità in questo passaggio. Per ulteriori informazioni, consulta Rete personalizzata per i pod.

Windows

1. Abilita l'assegnazione di prefissi IP.
   a. Apri `amazon-vpc-cni ConfigMap` per la modifica.

```
kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
```

   b. Aggiungi la seguente riga al file `data`:

```
  enable-windows-prefix-delegation: "true"
```

c. Salva il file e chiudi l'editor.

d. Accertati che la riga sia stata aggiunta a `ConfigMap`.

```
kubectl get configmap -n kube-system amazon-vpc-cni -o
 "jsonpath={.data.enable-windows-prefix-delegation}"
```

Se l'output restituito non è `true`, potrebbe essersi verificato un errore. Prova a completare di nuovo il passaggio.

> **Important**
>
> Anche se la sottorete dispone di indirizzi IP disponibili, se non dispone di blocchi `/28` contigui disponibili, verrà visualizzato il seguente errore negli eventi dei nodi:
>
> ```
> "failed to allocate a private IP/Prefix address:
>  InsufficientCidrBlocks: The specified subnet does not have enough
>  free cidr blocks to satisfy the request"
> ```
>
> Ciò può verificarsi a causa della frammentazione degli indirizzi IP secondari esistenti distribuiti in una sottorete. Per risolvere questo errore, creare una nuova sottorete e avviare i Pods lì oppure utilizzare una prenotazione CIDR della sottorete Amazon EC2 per riservare spazio all'interno di una sottorete da utilizzare con l'assegnazione del prefisso. Per ulteriori informazioni, consultare la sezione relativa a Prenotazioni della CIDR per la sottorete nella Guida per l'utente di Amazon VPC.

2. (Facoltativo) Specifica una configurazione aggiuntiva per controllare il comportamento di prescalabilità e scalabilità dinamica del cluster. Per ulteriori informazioni, vedere Opzioni di configurazione con la modalità di delega dei prefissi Windows attiva GitHub.

a. Apri `amazon-vpc-cni ConfigMap` per la modifica.

```
kubectl edit configmap -n kube-system amazon-vpc-cni -o yaml
```

b. Sostituisci *example values* con un valore maggiore di zero e aggiungi le voci necessarie alla sezione `data` di`ConfigMap`. Se imposti un valore per `warm-ip-target` o `minimum-ip-target`, il valore sostituisce qualsiasi valore impostato per `warm-prefix-target`.

```
warm-prefix-target: "1"
warm-ip-target: "5"
minimum-ip-target: "2"
```

c. Salva il file e chiudi l'editor.

3. Crea gruppi di nodi di Windows con almeno un tipo di istanza Nitro Amazon EC2. Per un elenco dei tipi di Nitro istanze, consulta Instances built on the Nitro System nella Amazon Amazon EC2 User Guide. Per impostazione predefinita, il numero massimo di Pods che è possibile distribuire su un nodo è pari a 110. Se vuoi aumentare o diminuire questo numero, specifica quanto segue nei dati utente per la configurazione di bootstrap. Sostituisci *max-pods-quantity* con il tuo valore di pod massimo.

```
-KubeletExtraArgs '--max-pods=max-pods-quantity'
```

Se stai distribuendo gruppi di nodi gestiti, questa configurazione deve essere aggiunta nel modello di avvio. Per ulteriori informazioni, consulta Personalizzazione di nodi gestiti con un modello di avvio. Per ulteriori informazioni sui parametri di configurazione dello script bootstrap di Windows, consulta Parametri di configurazione dello script di bootstrap.

2. Dopo aver implementato i nodi, visualizzare i nodi del cluster.

```
kubectl get nodes
```

Di seguito viene riportato un output di esempio:

```
NAME                                             STATUS     ROLES    AGE   VERSION
ip-192-168-22-103.region-code.compute.internal   Ready      <none>   19m   v1.XX.X-eks-6b7464
ip-192-168-97-94.region-code.compute.internal    Ready      <none>   19m   v1.XX.X-eks-6b7464
```

3. Descrivi uno dei nodi per determinare il valore di `max-pods` per il nodo e il numero di indirizzi IP disponibili. Sostituisci *192.168.30.193* con l'indirizzo `IPv4` nel nome di uno dei tuoi nodi restituiti nell'output del passaggio precedente.

```
kubectl describe node ip-192-168-30-193.region-code.compute.internal | grep 'pods\|
PrivateIPv4Address'
```

Di seguito viene riportato un output di esempio:

```
pods:                                       110
vpc.amazonaws.com/PrivateIPv4Address:  144
```

Nell'output precedente, `110` è il numero massimo di Pods che Kubernetes implementerà nel nodo, anche se sono disponibili *144* indirizzi IP.

Gruppi di sicurezza per Pods

I gruppi di sicurezza per i Pods integrano i gruppi di sicurezza Amazon EC2 con i Pods Kubernetes. È possibile utilizzare i gruppi di sicurezza Amazon EC2 per definire regole che consentono il traffico di rete in entrata e in uscita da e verso i Pods implementati nei nodi in esecuzione su molti tipi di istanza Amazon EC2 e Fargate. Per una descrizione dettagliata di questa funzionalità, consulta il post Introduzione ai gruppi di sicurezza per i podPods del blog.

Considerazioni

- Prima di implementare i gruppi di sicurezza per i Pods, considera i limiti e le condizioni seguenti:
- I gruppi di sicurezza per i Pods non possono essere utilizzati con i nodi Windows.
- I gruppi di sicurezza per i Pods possono essere utilizzati con cluster configurati per la famiglia `IPv6` che contengono i nodi Amazon EC2 utilizzando la versione 1.16.0 o successive del plug-in Amazon VPC CNI. Puoi utilizzare i gruppi di sicurezza per i Pods con cluster configurati per la famiglia `IPv6` che contengono solo nodi Fargate, utilizzando la versione 1.7.7 o successive del plugin Amazon VPC CNI. Per ulteriori informazioni, consulta `IPv6`indirizzi per cluster Pods e services
- I gruppi di sicurezza per Pods sono supportati dalla maggior parte delle famiglie di istanze Amazon EC2 basate su Nitro, ma non da tutte le generazioni di una famiglia. Ad esempio, vengono supportate la `m5` famiglia e le generazioni `r5` `m6g`c6g,,,, e `r6g` ad esempio. `c5` Non è supportato alcun tipo di istanza nella famiglia `t`. Per un elenco completo dei tipi di istanza supportati, consulta il file limits.go su GitHub. I nodi devono essere uno dei tipi di istanza elencati che contengono `IsTrunkingCompatible: true` nel file.

- Se utilizzi anche le policy di sicurezza dei Pod per limitare l'accesso alla mutazione del Pod, allora l'utente `eks:vpc-resource-controller` Kubernetes deve essere specificato in `ClusterRoleBinding` Kubernetes per il `role` a cui è assegnato il `psp`. Se si utilizza il `psp`, `role` e `ClusterRoleBinding` predefinito di Amazon EKS, questa è la `eks:podsecuritypolicy:authenticated` `ClusterRoleBinding`. Ad esempio, aggiungi l'utente alla sezione `subjects:`, come mostrato nell'esempio seguente:

```
[...]
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: system:authenticated
  - apiGroup: rbac.authorization.k8s.io
    kind: User
    name: eks:vpc-resource-controller
  - kind: ServiceAccount
    name: eks-vpc-resource-controller
```

- Se si utilizzano allo stesso tempo una rete personalizzata e i gruppi di sicurezza per i Pods, il gruppo di sicurezza specificato dai gruppi di sicurezza per i Pods viene utilizzato in alternativa al gruppo di sicurezza specificato nella `ENIConfig`.
- Se utilizzi la versione `1.10.2` o precedente del plug-in CNI di Amazon VPC e includi l'impostazione `terminationGracePeriodSeconds` nelle specifiche del Pod, il valore per l'impostazione non può essere zero.
- Se utilizzi la versione `1.10` o precedente del plug-in CNI di Amazon VPC o la versione `1.11` con l'impostazione predefinita `POD_SECURITY_GROUP_ENFORCING_MODE=strict`, i servizi Kubernetes di tipo `NodePort` e `LoadBalancer` che utilizzano le destinazioni di istanza con `externalTrafficPolicy` impostato su `Local` non sono supportati con i Pods a cui si assegnano i gruppi di sicurezza. Per ulteriori informazioni sull'utilizzo di un sistema di bilanciamento del carico con target di istanza, consulta la sezione Bilanciamento del carico di rete su Amazon EKS
- La fonte NAT è disabilitata per il traffico in uscita dai Pods con gruppi di sicurezza assegnati, in modo che vengano applicate le regole dei gruppi di sicurezza in uscita, se si utilizza la versione `1.10` o precedente del plug-in CNI di Amazon VPC o la versione `1.11` con l'impostazione predefinita `POD_SECURITY_GROUP_ENFORCING_MODE=strict`. Per accedere a Internet, è necessario avviare Pods con gruppi di sicurezza assegnati su nodi implementati in una sottorete privata configurata con un gateway NAT o un'istanza. I Podscon gruppi di sicurezza assegnati implementati nelle sottoreti pubbliche non sono in grado di accedere a Internet.

Se si utilizza la versione `1.11` o successiva del plug-in con `POD_SECURITY_GROUP_ENFORCING_MODE=standard`, il traffico Pod destinato all'esterno del VPC viene convertito nell'indirizzo IP dell'interfaccia di rete primaria dell'istanza. Per questo traffico vengono utilizzate le regole nei gruppi di sicurezza per l'interfaccia di rete primaria, anziché le regole nei gruppi di sicurezza dei Pod's.

- Per utilizzare le policy di rete Calico con Pods che hanno gruppi di sicurezza associati, dovrai utilizzare la versione `1.11.0` o successiva del plug-in CNI di Amazon VPC e impostare `POD_SECURITY_GROUP_ENFORCING_MODE=standard`. In caso contrario, i flussi di traffico da e verso i Pods con gruppi di sicurezza associati non sono soggetti all'applicazione della policy di rete Calico ma saranno limiti esclusivamente all'applicazione del gruppo di sicurezza di Amazon EC2. Per aggiornare la versione di CNI di Amazon VPC, consulta la sezione Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS
- I Pods in esecuzione su nodi Amazon EC2 che utilizzano gruppi di sicurezza in cluster che impiegano Nodelocal DNSCache sono supportati solo con la versione `1.11.0` o successiva del plug-in CNI di Amazon VPC e con `POD_SECURITY_GROUP_ENFORCING_MODE=standard`. Per aggiornare la versione del plug-in CNI di Amazon VPC, consulta la sezione Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS
- I gruppi di sicurezza per Pods potrebbero portare a un aumento della latenza di startup dei Pod per i Pods con un alto tasso di abbandono. Ciò è dovuto alla limitazione della velocità nel controller delle risorse.

Configurazione del Amazon VPC CNI plugin for Kubernetes per i gruppi di sicurezza per Pods

Per implementare i gruppi di sicurezza per i Pods

Se si utilizzano gruppi di sicurezza solo per i Pods di Fargate e non si dispone di nodi Amazon EC2 nel cluster, andare alla fase Implementare un'applicazione di esempio.

1. Controlla la versione del Amazon VPC CNI plugin for Kubernetes corrente con il comando seguente:

```
kubectl describe daemonset aws-node --namespace kube-system | grep amazon-k8s-cni:
 | cut -d : -f 3
```

   Di seguito viene riportato un output di esempio:

```
v1.7.6
```

Se la versione del Amazon VPC CNI plugin for Kubernetes è precedente alla `1.7.7`, aggiorna il plug-in alla versione `1.7.7` o successiva. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS

2. Aggiungi la policy IAM gestita `AmazonEKSVPCResourceController` al ruolo del cluster associato al cluster Amazon EKS. La policy consente al ruolo di gestire le interfacce di rete, i relativi indirizzi IP privati e i relativi allegati e distacchi da e verso le istanze di rete.

   a. Recupera il nome del ruolo IAM del cluster e archivialo in una variabile. Sostituisci *my-cluster* con il nome del cluster.

   ```
   cluster_role=$(aws eks describe-cluster --name my-cluster --query
    cluster.roleArn --output text | cut -d / -f 2)
   ```

   b. Collegare la policy al ruolo.

   ```
   aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
   AmazonEKSVPCResourceController --role-name $cluster_role
   ```

3. Abilitare il componente aggiuntivo CNI di Amazon VPC per gestire le interfacce di rete per i Pods impostando la variabile `ENABLE_POD_ENI` su `true` nel `aws-node` DaemonSet. Una volta impostato questo parametro su `true`, per ogni nodo nel cluster il componente aggiuntivo crea una risorsa cninode personalizzata. Il controller di risorse VPC crea e allega un'interfaccia di rete speciale chiamata interfaccia di rete trunk con la descrizione `aws-k8s-trunk-eni`.

   ```
   kubectl set env daemonset aws-node -n kube-system ENABLE_POD_ENI=true
   ```

   > **Note**
   >
   > L'interfaccia di rete trunk è inclusa nel numero massimo di interfacce di rete supportate dal tipo di istanza. Per un elenco del numero massimo di interfacce di rete supportate da ogni tipo di istanza, consulta Indirizzi IP per interfaccia di rete per tipo di istanza nella Amazon EC2 User Guide. Se il nodo ha già allegato il numero massimo di interfacce di rete standard, il controller di risorse VPC riserverà uno spazio. Sarà necessario ridurre i Pods in esecuzione in modo sufficiente da consentire al controller di scollegare ed

eliminare un'interfaccia di rete standard, creare l'interfaccia di rete trunk e allegarla all'istanza.

4. Con il seguente comando è possibile vedere quale nodo ha una risorsa `CNINode` personalizzata. Se viene restituito l'output `No resources found`, attendere alcuni secondi e riprovare. La fase precedente richiede il riavvio dei Pods Amazon VPC CNI plugin for Kubernetes, operazione che richiede alcuni secondi.

```
$ kubectl get cninode -A
     NAME FEATURES
     ip-192-168-64-141.us-west-2.compute.internal
 [{"name":"SecurityGroupsForPods"}]
     ip-192-168-7-203.us-west-2.compute.internal [{"name":"SecurityGroupsForPods"}]
```

Se si utilizzano versioni VPC CNI precedenti alla `1.15`, sono state utilizzate le etichette dei nodi anziché la risorsa `CNINode` personalizzata. Con il seguente comando, è possibile vedere quale nodo ha l'etichetta del nodo `aws-k8s-trunk-eni` impostata su `true`. Se viene restituito l'output `No resources found`, attendere alcuni secondi e riprovare. La fase precedente richiede il riavvio dei Pods Amazon VPC CNI plugin for Kubernetes, operazione che richiede alcuni secondi.

```
kubectl get nodes -o wide -l vpc.amazonaws.com/has-trunk-attached=true
-
```

Una volta creata l'interfaccia di rete del trunk, ai Pods possono essere assegnati indirizzi IP secondari dalla rete trunk o dalle interfacce di rete standard. L'interfaccia trunk viene eliminata automaticamente se il nodo viene eliminato.

Quando si implementa un gruppo di sicurezza per un Pod in una fase successiva, il controller di risorse VPC crea un'interfaccia di rete speciale denominata interfaccia di rete di filiali con una descrizione di `aws-k8s-branch-eni` e associa i gruppi di sicurezza. Le interfacce di rete di filiali vengono create in aggiunta alle interfacce di rete standard e trunk allegate al nodo.

Se si utilizzano probe liveness o readiness, è inoltre necessario disabilitare il demux precoce TCP, in modo che il `kubelet` possa connettersi a Pods su interfacce di rete di filiali tramite TCP. Per disabilitare TCP early demux, eseguire il comando seguente:

```
kubectl patch daemonset aws-node -n kube-system \
```

```
  -p '{"spec": {"template": {"spec": {"initContainers": [{"env":
[{"name":"DISABLE_TCP_EARLY_DEMUX","value":"true"}],"name":"aws-vpc-cni-
init"}]}}}}'
```

> **Note**
>
> Non è necessario eseguire il comando precedente se si utilizza la versione `1.11.0` o successive del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes e si imposta `POD_SECURITY_GROUP_ENFORCING_MODE=standard`, come descritto nella fase successiva.

5. Se il cluster utilizza `NodeLocal DNSCache`, se desideri utilizzare la policy di rete Calico con i tuoi Pods che dispongono dei propri gruppi di sicurezza, oppure se disponi di servizi Kubernetes di tipo `NodePort` e `LoadBalancer` che utilizzano le destinazioni di istanza con una `externalTrafficPolicy` impostata su `Local` per Pods a cui desideri assegnare i gruppi di sicurezza, dovrai utilizzare la versione `1.11.0` o successiva del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes e dovrai abilitare la seguente impostazione:

```
kubectl set env daemonset aws-node -n kube-system
 POD_SECURITY_GROUP_ENFORCING_MODE=standard
```

> **Important**
>
> - Le regole del gruppo di sicurezza per i Pod non si applicano al traffico tra Pods o tra Pods e services, ad esempio `kubelet` o `nodeLocalDNS` che si trovano sullo stesso nodo. I pod che utilizzano gruppi di sicurezza differenti sullo stesso nodo non possono comunicare perché sono configurati in sottoreti diverse e l'instradamento è disabilitato tra tali sottoreti.
> - Traffico in uscita dai Pods verso gli indirizzi esterni al VPC è l'indirizzo di rete convertito nell'indirizzo IP dell'interfaccia di rete primaria dell'istanza (a meno che non sia stato impostato `AWS_VPC_K8S_CNI_EXTERNALSNAT=true`). Per questo traffico vengono utilizzate le regole nei gruppi di sicurezza per l'interfaccia di rete primaria, anziché le regole nei gruppi di sicurezza dei Pod's.
> - Affinché questa impostazione venga applicata ai Pods esistenti, è necessario riavviare i Pods o i nodi su cui sono in esecuzione i Pods.

Implementare un'applicazione di esempio

Per utilizzare i gruppi di sicurezza per i Pods, è necessario disporre di un gruppo di sicurezza esistente ed è necessario implementare un `SecurityGroupPolicy` di Amazon EKS nel cluster, come descritto nella procedura seguente. I passaggi seguenti mostrano l'utilizzo delle policy di gruppo di sicurezza per un Pod. Se non diversamente indicato, completare tutti i passaggi nello stesso terminale, poiché nei passaggi successivi vengono utilizzate variabili che non persistono tra i terminali.

Per implementare un Pod di esempio con un gruppo di sicurezza

1. Crea uno spazio dei nomi Kubernetes in cui implementare le risorse. Puoi sostituire *my-namespace* con il nome di uno spazio dei nomi che desideri utilizzare.

```
kubectl create namespace my-namespace
```

2. Implementa una `SecurityGroupPolicy` Amazon EKS nel cluster.

   a. Copia i seguenti contenuti sul dispositivo. Puoi sostituire *podSelector* con **`serviceAccountSelector`** se preferisci selezionare i Pods in base alle etichette degli account di servizio. É necessario specificare uno selettore. Un `podSelector` vuoto (ad esempio, `podSelector: {}`) seleziona tutti i Pods nello spazio dei nomi. Puoi cambiare il *my-role* con il nome del tuo ruolo. Un `serviceAccountSelector` vuoto seleziona tutti gli account di servizio nello spazio dei nomi. Puoi sostituire *my-security-group-policy* con un nome per il tuo `SecurityGroupPolicy` e *my-namespace* con lo spazio dei nomi in cui desideri creare `SecurityGroupPolicy`.

   È necessario sostituire *my_pod_security_group_id* con l'ID di un gruppo di sicurezza esistente. Se non disponi ancora di un gruppo di sicurezza dovrai crearne uno. Per ulteriori informazioni, consulta Gruppi di sicurezza Amazon EC2 per le istanze Linux nella Guida per l'utente di Amazon EC2. Puoi specificare da 1 a 5 ID gruppo di sicurezza. Se si specificano più ID, la combinazione di tutte le regole in tutti i gruppi di sicurezza sarà valida per i Pods selezionati.

```
cat >my-security-group-policy.yaml <<EOF
apiVersion: vpcresources.k8s.aws/v1beta1
kind: SecurityGroupPolicy
metadata:
  name: my-security-group-policy
  namespace: my-namespace
```

```
spec:
  podSelector:
    matchLabels:
      role: my-role
  securityGroups:
    groupIds:
      - my_pod_security_group_id
EOF
```

Important

Il gruppo di sicurezza o i gruppi specificati per il Pod devono soddisfare i criteri seguenti:

- Devono essere esistenti. Se non lo sono, quando si implementa un Pod che corrisponde al selettore, il Pod rimane bloccato nel processo di creazione. Se si descrive il Pod, si otterrà un messaggio di errore simile al seguente: `An error occurred (InvalidSecurityGroupID.NotFound) when calling the CreateNetworkInterface operation: The securityGroup ID 'sg-05b1d815d1EXAMPLE' does not exist.`
- Devono consentire la comunicazione in ingresso dal gruppo di sicurezza applicato ai nodi (per `kubelet`) su tutte le porte per cui sono state configurate le sonde.
- Devono consentire la comunicazione in uscita tramite le porte 53 `TCP` e `UDP` a un gruppo di sicurezza assegnato ai Pods (o ai nodi su cui sono eseguiti i Pods) che eseguono CoreDNS. Il gruppo di sicurezza per i Pods CoreDNS deve consentire il traffico in ingresso tramite le porte 53 `TCP` e `UDP` dal gruppo di sicurezza specificato.
- Devono disporre delle regole in entrata e in uscita necessarie per comunicare con altri Pods.
- Devono disporre di regole che consentano ai Pods di comunicare con il piano di controllo di Kubernetes se utilizzi il gruppo di sicurezza con Fargate. Il modo più semplice per eseguire questa operazione consiste nello specificare il gruppo di sicurezza del cluster come uno dei gruppi di sicurezza.

Le policy dei gruppi di sicurezza si applicano solo ai nuovi Pods pianificati. Non incidono sui Pods in esecuzione.

b. Implementare la policy.

```
kubectl apply -f my-security-group-policy.yaml
```

3. Implementa un'applicazione di esempio con un'etichetta che corrisponda al valore *my-role* per *podSelector* specificato nel passaggio precedente.

a. Copia i seguenti contenuti sul dispositivo. Sostituisci i *valori di esempio* con i tuoi, quindi esegui il comando modificato. Se sostituisci il *my-role*, assicurati che sia uguale al valore che hai specificato per il selettore in un passaggio precedente.

```
cat >sample-application.yaml <<EOF
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-deployment
  namespace: my-namespace
  labels:
    app: my-app
spec:
  replicas: 4
  selector:
    matchLabels:
      app: my-app
  template:
    metadata:
      labels:
        app: my-app
        role: my-role
    spec:
      terminationGracePeriodSeconds: 120
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: my-app
  namespace: my-namespace
```

```
  labels:
    app: my-app
spec:
  selector:
    app: my-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
EOF
```

b. Implementare l'applicazione con il seguente comando. Quando si implementa l'applicazione, il Amazon VPC CNI plugin for Kubernetes corrisponde all'etichetta `role` e i gruppi di sicurezza specificati nel passaggio precedente vengono applicati al Pod.

```
kubectl apply -f sample-application.yaml
```

4. Visualizza i Pods implementati con l'applicazione di esempio. Nella parte restante di questo argomento, questo terminale è indicato come `TerminalA`.

```
kubectl get pods -n my-namespace -o wide
```

Di seguito viene riportato un output di esempio:

```
NAME                             READY   STATUS    RESTARTS   AGE     IP
       NODE                                            NOMINATED NODE   READINESS
 GATES
my-deployment-5df6f7687b-4fbjm   1/1     Running   0          7m51s   192.168.53.48
    ip-192-168-33-28.region-code.compute.internal   <none>           <none>
my-deployment-5df6f7687b-j9fl4   1/1     Running   0          7m51s
  192.168.70.145   ip-192-168-92-33.region-code.compute.internal   <none>
 <none>
my-deployment-5df6f7687b-rjxcz   1/1     Running   0          7m51s
  192.168.73.207   ip-192-168-92-33.region-code.compute.internal   <none>
 <none>
my-deployment-5df6f7687b-zmb42   1/1     Running   0          7m51s   192.168.63.27
    ip-192-168-33-28.region-code.compute.internal   <none>           <none>
```

> **Note**
>
> - Se dei Pods sono bloccati nello stato `Waiting`, esegui `kubectl describe pod` ***`my-deployment-xxxxxxxxxx-xxxxx`*** `-n` *`my-namespace`*. Se visualizzi `Insufficient permissions: Unable to create Elastic Network Interface.`, verifica di avere aggiunto la policy IAM al ruolo cluster IAM in un passaggio precedente.
> - Se dei Pods sono bloccati nello stato `Pending`, verifica che il tipo di istanza del nodo sia elencato in `limits.go` e che non sia già stato raggiunto il prodotto del numero massimo di interfacce di rete di filiali supportate dal tipo di istanza moltiplicato per il numero di nodi nel gruppo di nodi. Ad esempio, una istanza `m5.large` supporta nove interfacce di rete di filiali. Se il gruppo di nodi dispone di cinque nodi, è possibile creare un massimo di 45 interfacce di rete di filiali per il gruppo di nodi. Il 46° Pod che si tenta di implementare si troverà in uno stato di `Pending` finché non viene eliminato un altro Pod a cui sono associati gruppi di sicurezza.

Se si esegue `kubectl describe pod` ***`my-deployment-xxxxxxxxxx-xxxxx`*** **`-n`** ***`my-namespace`*** e si visualizza un messaggio simile al seguente, può essere ignorato in modo sicuro. Questo messaggio potrebbe essere visualizzato quando il Amazon VPC CNI plugin for Kubernetes tenta di impostare la rete host e fallisce durante la creazione dell'interfaccia di rete. Il plug-in registra questo evento fino a quando non viene creata l'interfaccia di rete.

```
Failed to create Pod sandbox: rpc error: code = Unknown desc = failed to set up
 sandbox container
 "e24268322e55c8185721f52df6493684f6c2c3bf4fd59c9c121fd4cdc894579f" network for Pod
 "my-deployment-5df6f7687b-4fbjm": networkPlugin
cni failed to set up Pod "my-deployment-5df6f7687b-4fbjm-c89wx_my-namespace"
 network: add cmd: failed to assign an IP address to container
```

Non è possibile superare il numero massimo di Pods eseguibili sul tipo di istanza. Per un elenco del numero massimo di Pods che è possibile eseguire su ogni tipo di istanza, consulta eni-max-pods.txt su GitHub. Quando si elimina un Pod a cui sono associati gruppi di sicurezza o si elimina il nodo su cui è in esecuzione il Pod, il controller di risorse VPC elimina l'interfaccia di rete di filiali. Se elimini un cluster con Pods che utilizzano Pods per gruppi di sicurezza, il controller non elimina le interfacce di rete delle diramazioni, per cui dovrai eliminarle

personalmente. Per informazioni su come eliminare le interfacce di rete, consulta Eliminare un'interfaccia di rete nella Guida per l'utente di Amazon EC2.

5. In un terminale separato, inserisci nella shell uno dei Pods. Nella parte restante di questo argomento, questo terminale è indicato come `TerminalB`. Sostituisci *`5df6f7687b-4fbjm`* con l'ID di uno dei Pods restituiti nell'output della fase precedente.

```
kubectl exec -it -n my-namespace my-deployment-5df6f7687b-4fbjm -- /bin/bash
```

6. Confermare il funzionamento dell'applicazione di esempio dalla shell in `TerminalB`.

```
curl my-app
```

Di seguito viene riportato un output di esempio:

```
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
[...]
```

L'output è stato ricevuto in quanto tutti i Pods che eseguono l'applicazione sono associati al gruppo di sicurezza creato. Tale gruppo contiene una regola che abilita il traffico tra tutti i Pods a cui è associato il gruppo di sicurezza. Il traffico DNS è consentito in uscita da tale gruppo di sicurezza al gruppo di sicurezza del cluster associato ai nodi. I nodi eseguono i Pods CoreDNS, su cui i Pods hanno eseguito la ricerca del nome.

7. Da `TerminalA`, rimuovere le regole del gruppo di sicurezza che consentono la comunicazione DNS dal gruppo di sicurezza al gruppo di sicurezza del cluster. Se in una fase precedente non hai aggiunto le regole DNS al gruppo di sicurezza del cluster, sostituisci *`$my_cluster_security_group_id`* con l'ID del gruppo di sicurezza in cui sono state create le regole.

```
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_tcp_rule_id
aws ec2 revoke-security-group-ingress --group-id $my_cluster_security_group_id --
security-group-rule-ids $my_udp_rule_id
```

8. Tentare di accedere nuovamente all'applicazione da `TerminalB`.

```
curl my-app
```

Di seguito viene riportato un output di esempio:

```
curl: (6) Could not resolve host: my-app
```

Il tentativo ha esito negativo perché il Pod non è più in grado di accedere ai Pods CoreDNS a cui è associato il gruppo di sicurezza del cluster. Il gruppo di sicurezza del cluster non dispone più delle relative regole che consentono la comunicazione DNS dal gruppo di sicurezza associato al Pod.

Se si tenta di accedere all'applicazione utilizzando gli indirizzi IP restituiti in una fase precedente per uno dei Pods, si riceve comunque una risposta perché tutte le porte sono consentite tra i Pods a cui è associato il gruppo di sicurezza e non è necessaria una ricerca del nome.

9. Dopo avere eseguito diverse prove, è possibile rimuovere la policy del gruppo di sicurezza, l'applicazione e il gruppo di sicurezza di esempio creati. Esegui i comandi seguenti da `TerminalA`.

```
kubectl delete namespace my-namespace
aws ec2 revoke-security-group-ingress --group-id $my_pod_security_group_id --
security-group-rule-ids $my_inbound_self_rule_id
wait
sleep 45s
aws ec2 delete-security-group --group-id $my_pod_security_group_id
```

Interfacce di rete multiple per Pods

Multus CNI è un plug-in CNI (Container Network Interface) per Amazon EKS che consente di allegare più interfacce di rete a un Pod. Per ulteriori informazioni, consulta la documentazione Multus-CNI su GitHub.

In Amazon EKS, ogni Pod ha un'interfaccia di rete assegnata dal plug-in CNI di Amazon VPC. Con Multus, è possibile creare un Pod multihomed con più interfacce. Ciò avviene grazie poiché Multus agisce come un "meta-plug-in"; un plug-in CNI che può chiamare altri plug-in CNI multipli. Il supporto AWS per Multus viene configurato con il plug-in CNI di Amazon VPC come plug-in delegato predefinito.

Considerazioni

- Amazon EKS non costruirà e pubblicherà i plug-in CNI della virtualizzazione I/O singola root (SR-IOV) e del Data Plane Development Kit (Piano Dati Kit Sviluppo, DPDK). Tuttavia, è possibile ottenere l'accelerazione dei pacchetti collegandosi direttamente ad Amazon EC2 Elastic Network Adapters (ENA) tramite il dispositivo host gestito Multus e plug-in `ipvlan`.
- Amazon EKS supporta Multus, che fornisce un processo generico che consente un semplice concatenamento di plug-in CNI aggiuntivi. Multus e il processo di concatenamento è supportato, ma AWS non fornisce supporto per tutti i plug-in CNI compatibili che possono essere concatenati o per problemi che possono sorgere in quei plug-in CNI non correlati alla configurazione del concatenamento.
- Amazon EKS fornisce supporto e gestione del ciclo di vita per il plug-in Multus, ma non è responsabile di alcun indirizzo IP o gestione aggiuntiva associata alle interfacce di rete aggiuntive. L'indirizzo IP e la gestione dell'interfaccia di rete predefinita che utilizza il plug-in CNI di Amazon VPC rimangono invariati.
- Solo il plug-in CNI di Amazon VPC è ufficialmente supportato come plug-in delegato predefinito. Se si sceglie di non utilizzare il plug-in CNI di Amazon VPC per la rete primaria, è necessario modificare il manifesto di installazione Multus pubblicato per riconfigurare il plug-in delegato predefinito in un CNI alternativo.
- Multus è supportato solo quando si utilizza il CNI di Amazon VPC come CNI principale. Non supportiamo il CNI di Amazon VPC se utilizzato per interfacce di ordine superiore, secondarie o di altro tipo.
- Per evitare che il plug-in CNI di Amazon VPC tenti di gestire interfacce di rete aggiuntive assegnate ai Pods, aggiungi il seguente tag alle interfacce di rete:

  chiave: `node.k8s.amazonaws.com/no_manage`

  valore: `true`
- Multus è compatibile con policy di rete, ma queste devono essere arricchite per includere porte e indirizzi IP che possono far parte di interfacce di rete aggiuntive allegate ai Pods.

Per una procedura di implementazione, vedere la Guida all'installazione di Multus su GitHub.

Plugin CNI compatibili alternativi

Amazon VPC CNI plugin for Kubernetes è l'unico plug-in CNI supportato da Amazon EKS. Amazon EKS esegue Kubernetes a monte, permettendo di installare plug-in CNI compatibili alternativi sui

nodi Amazon EC2 del cluster. Se nel cluster sono presenti nodi Fargate, Amazon VPC CNI plugin for Kubernetes è già presente sui nodi Fargate. È l'unico plug-in CNI che puoi utilizzare con i nodi Fargate. Qualsiasi tentativo di installare un plug-in CNI alternativo sui nodi Fargate avrà esito negativo.

Se prevedi di utilizzare un plug-in CNI alternativo sui nodi Amazon EC2, ti consigliamo vivamente di ottenere supporto commerciale o di disporre delle competenze interne per risolvere i problemi e contribuire alle correzioni del progetto del plug-in CNI.

Amazon EKS dispone di una rete di partner che offrono supporto per plug-in CNI compatibili alternativi. Per informazioni dettagliate sulle versioni, sulle qualifiche e sui test eseguiti, consulta la documentazione dei partner riportata di seguito.

Partner	Product	Documentazione
Tigera	Calico	Istruzioni di installazione
Isovalent	Cilium	Istruzioni di installazione
Juniper	Rete Contrail cloud native (CN2)	Istruzioni di installazione
VMware	Antrea	Istruzioni di installazione

Amazon EKS mira a dare all'utente una vasta gamma di opzioni per coprire tutti i casi d'uso.

Plugin alternativi compatibili per le politiche di rete

Calico è una soluzione ampiamente adottata per il networking e la sicurezza dei container. L'utilizzo Calico di EKS offre un'applicazione delle politiche di rete completamente conforme per i cluster EKS. Inoltre, puoi scegliere Calico di utilizzare la rete, che conserva gli indirizzi IP del VPC sottostante. Calico Cloud migliora le funzionalità diCalico Open Source, fornendo funzionalità avanzate di sicurezza e osservabilità.

Che cosa è la AWS Load Balancer Controller?

AWS Load Balancer ControllerGestisce AWS Elastic Load Balancers per un cluster. Kubernetes Puoi utilizzare il controller per esporre le app del cluster a Internet. Il controller fornisce sistemi di

AWS bilanciamento del carico che puntano alle risorse Cluster Service o Ingress. In altre parole, il controller crea un unico indirizzo IP o nome DNS che punta a più pod del cluster.

Il controller controlla le nostre risorse. Kubernetes Ingress Service In risposta, crea le risorse AWS Elastic Load Balancing appropriate. È possibile configurare il comportamento specifico dei sistemi di bilanciamento del carico applicando annotazioni alle risorse. Kubernetes Ad esempio, è possibile collegare gruppi di AWS sicurezza ai sistemi di bilanciamento del carico utilizzando le annotazioni.

Il controller fornisce le risorse seguenti:

Kubernetes `Ingress`

LBC crea un AWS Application Load Balancer (ALB) quando si crea un. Kubernetes `Ingress` Esamina le annotazioni che puoi applicare a una risorsa Ingress.

Servizio Kubernetes del tipo `LoadBalancer`

L'LBC crea un AWS Network Load Balancer (NLB) quando si crea un Kubernetes servizio di tipo. `LoadBalancer` Esamina le annotazioni che puoi applicare a una risorsa del servizio.

In passato, il Kubernetes network load balancer veniva utilizzato per le destinazioni di esempio, ma l'LBC veniva utilizzato per le destinazioni IP. Con AWS Load Balancer Controller versione `2.3.0` o successiva, puoi creare NLB utilizzando entrambi i tipi di destinazione. Per ulteriori informazioni sui tipi di destinazioni dei NLB, consultare Target type (Tipo di destinazione) nella Guida per l'utente di Network Load Balancer.

Il controller è un progetto open source gestito su. GitHub

Prima di implementare il controller, è consigliabile esaminare i prerequisiti e le considerazioni in Bilanciamento del carico di applicazione su Amazon EKS e Bilanciamento del carico di rete su Amazon EKS. In questi argomenti, verrà distribuita un'app di esempio che include un sistema di AWS bilanciamento del carico.

Installa il controller

- Scopri come farlothe section called “Installa con Helm”.Utilizza questa procedura se non conosci Amazon EKS. Questa procedura utilizza Helm, un gestore di pacchetti per Kubernetes e semplifica `eksctl`l'installazione di LBC.

- In alternativa,. the section called “Installa con Manifests” Questa procedura è appropriata per configurazioni di cluster avanzate. Ciò include i cluster con accesso di rete limitato ai registri di container pubblici.

Esegui la migrazione da versioni di controller obsolete

- Se disponi di versioni obsolete di quelle installate, scoprite come farlo. AWS Load Balancer Controller the section called “Migrazione da un controller obsoleto”
- Le versioni obsolete non possono essere aggiornate. È necessario rimuoverle e installarne una versione aggiornata. AWS Load Balancer Controller
- Le versioni obsolete includono:
  - AWS ALB Ingress Controller for Kubernetes («Ingress Controller»), un predecessore del. AWS Load Balancer Controller
  - Qualsiasi versione di `0.1.`*x* AWS Load Balancer Controller

Fornitore di servizi cloud legacy

Kubernetesinclude un provider cloud legacy per AWS. Il provider di cloud legacy è in grado di fornire sistemi di bilanciamento del AWS carico, in modo simile a. AWS Load Balancer Controller Il provider cloud legacy crea Classic Load Balancer. Se non si installaAWS Load Balancer Controller, per impostazione predefinita Kubernetes utilizzerà il provider cloud legacy. È necessario installare AWS Load Balancer Controller ed evitare di utilizzare il provider di cloud legacy.

> **Important**
>
> Nelle versioni 2.5 e successive, AWS Load Balancer Controller diventa il controller predefinito per le risorse di Kubernetes servizio con `type: LoadBalancer` e crea un AWS Network Load Balancer (NLB) per ogni servizio. Lo fa creando un webhook mutante per i servizi, che imposta il campo `spec.loadBalancerClassa service.k8s.aws/nlb` per nuovi servizi di`type: LoadBalancer`. È possibile disattivare questa funzione e tornare a utilizzare il Cloud Provider obsoleto come controller predefinito, impostando il valore della tabella di comando `enableServiceMutatorWebhook` a `false`. Il cluster non eseguirà il provisioning dei nuovi Classic Load Balancer per i tuoi servizi a meno che non disattivi questa funzionalità. I sistemi Classic Load Balancer esistenti continueranno a funzionare.

Installa AWS Load Balancer Controller usando Helm

Questo argomento descrive come installare AWS Load Balancer Controller utilizzando Helm, un gestore di pacchetti per Kubernetes e. `eksctl` Il controller viene installato con le opzioni predefinite. Per ulteriori informazioni sul controller, inclusi i dettagli sulla configurazione con annotazioni, consulta la AWS Load Balancer Controllerdocumentazione su. GitHub

Nei passaggi che seguono, sostituire *`example values`* con i propri valori.

Prerequisiti

Prima di iniziare questo tutorial, è necessario installare e configurare i seguenti strumenti e risorse necessarie per creare e gestire un cluster Amazon EKS.

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.
- Un provider AWS Identity and Access Management (IAM) OpenID Connect (OIDC) esistente per il tuo cluster. Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- Assicurati che Amazon VPC CNI plugin for Kubernetes, `kube-proxy` e i componenti aggiuntivi CoreDNS siano alle versioni minime elencate in Token dell'account di servizio.
- Familiarità con AWS Elastic Load Balancing. Per ulteriori informazioni, consulta la Guida per l'utente di Elastic Load Balancing.
- Familiarità con il servizio Kubernetes e le risorse in ingresso.

- Helm installato localmente.

Fase 1: Creare un ruolo IAM utilizzando **`eksctl`**

> **Note**
>
> Devi solo creare un ruolo IAM per ogni AWS account. AWS Load Balancer Controller Controlla se `AmazonEKSLoadBalancerControllerRole` esiste nella console IAM. Se questo ruolo esiste, passa athe section called “Fase 2: Installazione AWS Load Balancer Controller”.

Creare una policy IAM

1. Scaricare una policy IAM per il AWS Load Balancer Controller che consente di effettuare chiamate alle API AWS per conto dell'utente.

   AWS

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy_us-gov.json
   ```

   ```
   $ mv iam_policy_us-gov.json iam_policy.json
   ```

2. Creare una policy IAM utilizzando le policy scaricate nel passaggio precedente.

   ```
   $ aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
       --policy-document file://iam_policy.json
   ```

   > **Note**
   >
   > Se si visualizza la politica in AWS Management Console, la console mostra gli avvisi per il servizio ELB, ma non per il servizio ELB v2. Ciò accade perché alcune azioni nella policy esistono per ELB v2, ma non per ELB. Queste avvertenze per ELB possono essere ignorate.

Crea un ruolo IAM utilizzando **`eksctl`**

- Sostituisci *`my-cluster`* con il nome del cluster e *`111122223333`* con il tuo ID account, quindi esegui il comando. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, `arn:aws:` sostituiscilo con. `arn:aws-us-gov:`

  ```
  $ eksctl create iamserviceaccount \
  ```

```
  --cluster=my-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-
arn=arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```

Fase 2: Installazione AWS Load Balancer Controller

Installazione AWS Load Balancer Controller tramite Helm V3

1. Aggiungi l'archivio cartografico `eks-charts` Helm. AWS mantiene questo repository attivo. GitHub

```
$ helm repo add eks https://aws.github.io/eks-charts
```

2. Aggiornare il repository locale per assicurarsi di avere i grafici più recenti.

```
$ helm repo update eks
```

3. Installa AWS Load Balancer Controller.

   Sostituisci *my-cluster* con il nome del cluster. Nel comando seguente, `aws-load-balancer-controller` indica l'account del servizio Kubernetes creato in una fase precedente.

   Per ulteriori informazioni sulla configurazione del diagramma di timoneria, vedere on. `values.yaml` GitHub

```
$ helm install aws-load-balancer-controller eks/aws-load-balancer-controller \
  -n kube-system \
  --set clusterName=my-cluster \
  --set serviceAccount.create=false \
  --set serviceAccount.name=aws-load-balancer-controller
```

   a. Se stai implementando il controller su nodi Amazon EC2 che hanno accesso limitato ad Amazon EC2 Instance Metadata Service (IMDS) o in Fargate, aggiungi questi flag al comando `helm` seguente:

      - **--set region=*region-code***

- **`--set vpcId=`*`vpc-xxxxxxxx`***

b. Per visualizzare le versioni disponibili di Helm Chart e Load Balancer Controller, utilizzate il seguente comando:

```
helm search repo eks/aws-load-balancer-controller --versions
```

> **Important**
>
> Il grafico implementato non riceve automaticamente gli aggiornamenti di sicurezza. È necessario eseguire manualmente l'aggiornamento a un grafico più recente quando diventa disponibile. Durante l'aggiornamento, passate *`install`* al comando precedente**`upgrade`**.
> Il `helm install` comando installa automaticamente le definizioni di risorse personalizzate (CRDs) per il controller. Il `helm upgrade` comando non lo fa. Se si utilizza`helm upgrade`, , è necessario installare manualmente ilCRDs. Esegui il comando seguente per installareCRDs:
>
> ```
> wget https://raw.githubusercontent.com/aws/eks-charts/master/stable/aws-load-balancer-controller/crds/crds.yaml
> kubectl apply -f crds.yaml
> ```

Passaggio 3: Verificare che il controller sia installato

1. Verificare che il controller sia installato.

```
$ kubectl get deployment -n kube-system aws-load-balancer-controller
```

Di seguito viene riportato un output di esempio:

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

Riceverai l'output precedente se hai eseguito l'implementazione con Helm. Se hai eseguito l'implementazione utilizzando il manifesto Kubernetes, hai una sola replica.

2. Prima di utilizzare il controller per il provisioning AWS delle risorse, il cluster deve soddisfare requisiti specifici. Per ulteriori informazioni, consultare Bilanciamento del carico di applicazione su Amazon EKS e Bilanciamento del carico di rete su Amazon EKS.

Installa il AWS Load Balancer Controller componente aggiuntivo utilizzando Manifests Kubernetes

Questo argomento descrive come installare il controller scaricando e applicando i Kubernetes manifesti. É possibile visualizzare l'intera documentazione per il controller su GitHub.

Nei passaggi che seguono, sostituire *example values* con i propri valori.

Prerequisiti

Prima di iniziare questo tutorial, è necessario installare e configurare i seguenti strumenti e risorse necessarie per creare e gestire un cluster Amazon EKS.

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.
- Un provider AWS Identity and Access Management (IAM) OpenID Connect (OIDC) esistente per il cluster. Per determinare se disponi già di un provider IAM o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
- Assicurati che Amazon VPC CNI plugin for Kubernetes, `kube-proxy` e i componenti aggiuntivi CoreDNS siano alle versioni minime elencate in Token dell'account di servizio.
- Familiarità con AWS Elastic Load Balancing. Per ulteriori informazioni, consulta la Guida per l'utente di Elastic Load Balancing.
- Familiarità con il servizio Kubernetes e le risorse in ingresso.

Fase 1: Configurare IAM

> **Note**
>
> Devi solo creare un ruolo IAM per ogni AWS account. AWS Load Balancer Controller Controlla se `AmazonEKSLoadBalancerControllerRole` esiste nella console IAM. Se questo ruolo esiste, passa athe section called “Fase 2: Installazione `cert-manager`”.

Creare una policy IAM

1. Scaricare una policy IAM per il AWS Load Balancer Controller che consente di effettuare chiamate alle API AWS per conto dell'utente.

   AWS

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy.json
   ```

   AWS GovCloud (US)

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/install/iam_policy_us-gov.json
   ```

   ```
   $ mv iam_policy_us-gov.json iam_policy.json
   ```

2. Creare una policy IAM utilizzando le policy scaricate nel passaggio precedente.

   ```
   $ aws iam create-policy \
       --policy-name AWSLoadBalancerControllerIAMPolicy \
       --policy-document file://iam_policy.json
   ```

   > **Note**
   >
   > Se si visualizza la politica in AWS Management Console, la console mostra gli avvisi per il servizio ELB, ma non per il servizio ELB v2. Ciò accade perché alcune azioni nella policy esistono per ELB v2, ma non per ELB. Queste avvertenze per ELB possono essere ignorate.

eksctl

Crea un ruolo IAM utilizzando **`eksctl`**

- Sostituisci *`my-cluster`* con il nome del cluster e *`111122223333`* con il tuo ID account, quindi esegui il comando. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, `arn:aws:` sostituiscilo con. `arn:aws-us-gov:`

```
$ eksctl create iamserviceaccount \
  --cluster=my-cluster \
  --namespace=kube-system \
  --name=aws-load-balancer-controller \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --attach-policy-
arn=arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --approve
```

AWS CLI and kubectl

Crea un ruolo IAM utilizzando e AWS CLI**kubectl**

1. Recupera gli ID del fornitore OIDC del cluster e archivialo in una variabile.

```
oidc_id=$(aws eks describe-cluster --name my-cluster --query
 "cluster.identity.oidc.issuer" --output text | cut -d '/' -f 5)
```

2. Determina se un fornitore OIDC IAM con l'ID del tuo cluster è già presente nel tuo account. È necessario OIDC configurarlo sia per il cluster che per IAM.

```
aws iam list-open-id-connect-providers | grep $oidc_id | cut -d "/" -f4
```

   Se viene restituito un output, devi creare un provider OIDC IAM per il tuo cluster. Se non viene restituito alcun output, devi creare un provider IAM OIDC per il cluster. Per ulteriori informazioni, consulta Crea un OIDC provider IAM per il tuo cluster.

3. Copia i seguenti contenuti sul dispositivo. Sostituisci *111122223333* con l'ID del tuo account. Sostituiscilo *region-code* con Regione AWS quello in cui si trova il cluster. Sostituisci *EXAMPLED539D4633E53DE1B71EXAMPLE* con l'output restituito nella fase precedente. Se il cluster si trova negli AWS GovCloud Stati Uniti orientali o AWS GovCloud negli Stati Uniti occidentali Regioni AWS, `arn:aws:` sostituiscilo con. `arn:aws-us-gov:` Dopo aver sostituito il testo, esegui il comando modificato per creare il file `load-balancer-role-trust-policy.json`.

```
cat >load-balancer-role-trust-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": {
                "Federated": "arn:aws:iam::111122223333:oidc-provider/
oidc.eks.region-code.amazonaws.com/id/EXAMPLED539D4633E53DE1B71EXAMPLE"
            },
            "Action": "sts:AssumeRoleWithWebIdentity",
            "Condition": {
                "StringEquals": {
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:aud": "sts.amazonaws.com",
                    "oidc.eks.region-code.amazonaws.com/
id/EXAMPLED539D4633E53DE1B71EXAMPLE:sub": "system:serviceaccount:kube-
system:aws-load-balancer-controller"
                }
            }
        }
    ]
}
EOF
```

4. Crea il ruolo IAM.

```
aws iam create-role \
  --role-name AmazonEKSLoadBalancerControllerRole \
  --assume-role-policy-document file://"load-balancer-role-trust-policy.json"
```

5. Allegare la policy IAM gestita da Amazon EKS richiesta al ruolo IAM. Sostituisci *111122223333* con l'ID del tuo account.

```
aws iam attach-role-policy \
  --policy-arn
 arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerIAMPolicy \
  --role-name AmazonEKSLoadBalancerControllerRole
```

6. Copia i seguenti contenuti sul dispositivo. Sostituisci *111122223333* con l'ID del tuo account. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali) Regioni AWS, sostituiscilo con. `arn:aws:` `arn:aws-us-gov:` Dopo aver sostituito il testo, esegui il comando modificato per creare il file `aws-load-balancer-controller-service-account.yaml`.

```
cat >aws-load-balancer-controller-service-account.yaml <<EOF
```

```
apiVersion: v1
kind: ServiceAccount
metadata:
  labels:
    app.kubernetes.io/component: controller
    app.kubernetes.io/name: aws-load-balancer-controller
  name: aws-load-balancer-controller
  namespace: kube-system
  annotations:
    eks.amazonaws.com/role-arn:
 arn:aws:iam::111122223333:role/AmazonEKSLoadBalancerControllerRole
EOF
```

7. Creare l'account di servizio Kubernetes sul cluster. L'account del servizio Kubernetes denominato `aws-load-balancer-controller` è annotato con il ruolo IAM creato precedentemente e denominato *`AmazonEKSLoadBalancerControllerRole`*.

```
$ kubectl apply -f aws-load-balancer-controller-service-account.yaml
```

Fase 2: Installazione **`cert-manager`**

Installa `cert-manager` utilizzando uno dei metodi seguenti per inserire la configurazione del certificato nei Webhook. Per ulteriori informazioni, consulta Getting Started nella documentazione *`cert-manager`*.

Si consiglia di utilizzare il registro dei `quay.io` contenitori per l'installazione`cert-manager`. Se i tuoi nodi non hanno accesso al registro dei `quay.io` container, installa `cert-manager` utilizzando Amazon ECR (vedi sotto).

Quay.io

Installa usando Quay.io **`cert-manager`**

- Se i tuoi nodi hanno accesso al registro del container `quay.io`, installa `cert-manager` per inserire la configurazione del certificato nei Webhook.

```
$ kubectl apply \
    --validate=false \
    -f https://github.com/jetstack/cert-manager/releases/download/v1.13.5/cert-
manager.yaml
```

Amazon ECR

Installazione **`cert-manager`** tramite Amazon ECR

1. Installa `cert-manager` utilizzando uno dei metodi seguenti per inserire la configurazione del certificato nei Webhook. Per ulteriori informazioni, consulta Getting Started nella documentazione *`cert-manager`*.
2. Eseguire il download del manifesto.

```
curl -Lo cert-manager.yaml https://github.com/jetstack/cert-manager/releases/
download/v1.13.5/cert-manager.yaml
```

3. Estrai le seguenti immagini e inviale a un repository a cui i tuoi nodi hanno accesso. Per ulteriori informazioni su come estrarre, taggare e inviare immagini al tuo repository, consulta la sezione Copia di un'immagine di container da un repository a un altro.

```
quay.io/jetstack/cert-manager-cainjector:v1.13.5
quay.io/jetstack/cert-manager-controller:v1.13.5
quay.io/jetstack/cert-manager-webhook:v1.13.5
```

4. Nel manifesto, sostituisci `quay.io` per le tre immagini con il nome del tuo registro. Il comando seguente presuppone che il nome del repository privato sia lo stesso del repository di origine. Sostituisci *`111122223333.dkr.ecr.region-code.amazonaws.com`* con il tuo registro privato.

```
$ sed -i.bak -e 's|quay.io|111122223333.dkr.ecr.region-code.amazonaws.com|' ./
cert-manager.yaml
```

5. Applicare il file manifesto.

```
$ kubectl apply \
    --validate=false \
    -f ./cert-manager.yaml
```

Fase 3: Installazione AWS Load Balancer Controller

Installazione AWS Load Balancer Controller tramite un Kubernetes manifesto

1. Scaricare le specifiche del controller. Per ulteriori informazioni sull'utilizzo del controller, consulta la documentazione su GitHub.

```
curl -Lo v2_7_2_full.yaml https://github.com/kubernetes-sigs/aws-load-balancer-controller/releases/download/v2.7.2/v2_7_2_full.yaml
```

2. Apporta le seguenti modifiche al file.

   a. Se hai scaricato il file `v2_7_2_full.yaml`, emetti il seguente comando per rimuovere la sezione `ServiceAccount` nel manifesto. Se non rimuovi questa sezione, l'annotazione richiesta apportata all'account del servizio in un passaggio precedente viene sovrascritta. Inoltre, se elimini il controller, conserva l'account del servizio creato in una fase precedente.

   ```
   $ sed -i.bak -e '596,604d' ./v2_7_2_full.yaml
   ```

   Se hai scaricato una versione diversa del file, apri il file in un editor e rimuovi le righe seguenti.

   ```
   apiVersion: v1
   kind: ServiceAccount
   metadata:
     labels:
       app.kubernetes.io/component: controller
       app.kubernetes.io/name: aws-load-balancer-controller
     name: aws-load-balancer-controller
     namespace: kube-system
   ---
   ```

   b. Sostituisci `your-cluster-name` nella sezione `Deployment spec` del file con il nome del cluster sostituendo *`my-cluster`* con il nome del tuo cluster.

   ```
   $ sed -i.bak -e 's|your-cluster-name|my-cluster|' ./v2_7_2_full.yaml
   ```

   c. Se i tuoi nodi non hanno accesso ai repository di immagini Amazon EKS Amazon ECR, devi estrarre la seguente immagine e inviarla a un repository a cui i tuoi nodi hanno accesso.

Per ulteriori informazioni su come estrarre, taggare e inviare un'immagine al tuo repository, consulta la sezione Copia di un'immagine di container da un repository a un altro.

```
public.ecr.aws/eks/aws-load-balancer-controller:v2.7.2
```

Aggiungi il nome del tuo registro al manifesto. Il comando seguente presuppone che il nome del repository privato sia lo stesso del repository di origine e aggiunge il nome del tuo registro privato al file. Sostituisci *111122223333.dkr.ecr.region-code.amazonaws.com* con il tuo registro. Questa riga presuppone che il nome del repository privato sia lo stesso del repository di origine. In caso contrario, sostituisci il testo `eks/aws-load-balancer-controller` dopo il nome del registro privato con il nome del tuo repository.

```
$ sed -i.bak -e 's|public.ecr.aws/eks/aws-load-balancer-
controller|111122223333.dkr.ecr.region-code.amazonaws.com/eks/aws-load-
balancer-controller|' ./v2_7_2_full.yaml
```

d. (Richiesto solo per Fargate o IMDS con restrizioni)

Se stai eseguendo l'implementazione del controller su nodi Amazon EC2 che hanno accesso limitato al servizio di metadati dell'istanza Amazon EC2 (IMDS) o su Fargate, aggiungi i **`following parameters`** in `- args:`.

```
[...]
spec:
      containers:
        - args:
            - --cluster-name=your-cluster-name
            - --ingress-class=alb
            - --aws-vpc-id=vpc-xxxxxxxx
            - --aws-region=region-code


[...]
```

3. Applicare il file.

```
$ kubectl apply -f v2_7_2_full.yaml
```

4. Scarica il manifesto di `IngressClass` e `IngressClassParams` sul cluster.

```
$ curl -Lo v2_7_2_ingclass.yaml https://github.com/kubernetes-sigs/aws-load-balancer-controller/releases/download/v2.7.2/v2_7_2_ingclass.yaml
```

5. Applica il manifesto al cluster.

```
$ kubectl apply -f v2_7_2_ingclass.yaml
```

Passo 4: Verificare che il controller sia installato

1. Verificare che il controller sia installato.

```
$ kubectl get deployment -n kube-system aws-load-balancer-controller
```

Di seguito viene riportato un output di esempio:

```
NAME                           READY   UP-TO-DATE   AVAILABLE   AGE
aws-load-balancer-controller   2/2     2            2           84s
```

Riceverai l'output precedente se hai eseguito l'implementazione con Helm. Se hai eseguito l'implementazione utilizzando il manifesto Kubernetes, hai una sola replica.

2. Prima di utilizzare il controller per il provisioning AWS delle risorse, il cluster deve soddisfare requisiti specifici. Per ulteriori informazioni, consultare Bilanciamento del carico di applicazione su Amazon EKS e Bilanciamento del carico di rete su Amazon EKS.

Migrazione da un controller obsoleto

Questo argomento descrive come migrare da versioni di controller obsolete. Più specificamente, descrive come rimuovere le versioni obsolete di. AWS Load Balancer Controller

- Le versioni obsolete non possono essere aggiornate. Devono essere rimosse e deve essere installata una versione corrente di LBC.
- Le versioni obsolete includono:
  - AWS ALB Ingress Controller for Kubernetes («Ingress Controller»), un predecessore del. AWS Load Balancer Controller
  - Qualsiasi versione di `0.1.`*x* AWS Load Balancer Controller

Rimuovi la versione obsoleta del controller

> **Note**
>
> È possibile che la versione obsoleta sia stata installata utilizzando Helm o manualmente con i manifest. Kubernetes Completare la procedura utilizzando lo strumento con cui è stata installata in origine.

Rimuovi Ingress Controller usando Helm

1. Se il grafico Helm `incubator/aws-alb-ingress-controller` è installato, procedi alla disinstallazione.

   ```
   $ helm delete aws-alb-ingress-controller -n kube-system
   ```

2. Se la versione `0.1.x` del grafico `eks-charts/aws-load-balancer-controller` è installata, procedi alla disinstallazione. L'aggiornamento dalla versione `0.1.x` alla versione `1.0.0` non ha esito positivo a causa di incompatibilità con la versione dell'API webhook.

   ```
   $ helm delete aws-load-balancer-controller -n kube-system
   ```

Rimuovi Ingress Controller usando manifest Kubernetes

1. Verificare che il controller sia già installato.

   ```
   $ kubectl get deployment -n kube-system alb-ingress-controller
   ```

   Questo è l'output restituito se il controller non è installato.

   Errore dal server (NotFound): deployments.apps "" non trovato alb-ingress-controller

   Questo è l'output restituito se il controller è installato.

   ```
   NAME                    READY UP-TO-DATE AVAILABLE AGE
   alb-ingress-controller 1/1   1          1         122d
   ```

2. Immettere i seguenti comandi per rimuovere il controller.

```
$ kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-
ingress-controller/v1.1.8/docs/examples/alb-ingress-controller.yaml
kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-alb-
ingress-controller/v1.1.8/docs/examples/rbac-role.yaml
```

Migrazione ad AWS Load Balancer Controller

Per migrare dall'ALB Ingress Controller a, devi: Kubernetes AWS Load Balancer Controller

1. Rimuovere l'ALB Ingress Controller (vedi sopra).
2. Installa il. AWS Load Balancer Controller
3. Aggiungi una policy aggiuntiva al ruolo IAM utilizzato da LBC. Questa policy consente a LBC di gestire le risorse create dall'ALB Ingress Controller per. Kubernetes

Aggiungi la politica di migrazione al ruolo IAM. AWS Load Balancer Controller

1. Scaricare la policy IAM. Questa policy consente a LBC di gestire le risorse create dall'ALB Ingress Controller per. Kubernetes É anche possibile consultare la policy.

   ```
   $ curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-
   controller/v2.7.2/docs/install/iam_policy_v1_to_v2_additional.json
   ```

2. Se il tuo cluster si trova negli AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali), sostituiscilo con. Regioni AWS`arn:aws:` `arn:aws-us-gov:` . `arn:aws:`

   ```
   $ sed -i.bak -e 's|arn:aws:|arn:aws-us-gov:|' iam_policy_v1_to_v2_additional.json
   ```

3. Creare la policy IAM e annotate l'ARN restituito.

   ```
   $ aws iam create-policy \
     --policy-name AWSLoadBalancerControllerAdditionalIAMPolicy \
     --policy-document file://iam_policy_v1_to_v2_additional.json
   ```

4. Allega la policy IAM al ruolo IAM utilizzato da LBC. Sostituisci *`your-role-name`* con il nome del ruolo, ad esempio`AmazonEKSLoadBalancerControllerRole`.

   Se hai creato il ruolo utilizzando`eksctl`, quindi per trovare il nome del ruolo che è stato creato, apri la AWS CloudFormation console e seleziona lo stack eksctl- ***`my-cluster`*** - -. addon-

iamserviceaccount-kube-system aws-load-balancer-controller Selezionare la scheda Risorse. Il nome del ruolo è nella colonna ID fisico. Se il tuo cluster si trova nello AWS GovCloud (Stati Uniti orientali) o AWS GovCloud (Stati Uniti occidentali), sostituiscilo con. Regioni AWS`arn:aws:` `arn:aws-us-gov:`

```
$ aws iam attach-role-policy \
  --role-name your-role-name \
  --policy-arn
 arn:aws:iam::111122223333:policy/AWSLoadBalancerControllerAdditionalIAMPolicy
```

Utilizzo del componente aggiuntivo CoreDNS di Amazon EKS

CoreDNS è un server DNS flessibile ed estensibile che può fungere da DNS del cluster Kubernetes. Quando si avvia un cluster Amazon EKS con almeno un nodo, due repliche dell'immagine CoreDNS vengono implementate per impostazione predefinita, a prescindere dal numero di nodi implementati nel cluster. I Pods CoreDNS forniscono la risoluzione dei nomi per tutti i Pods del cluster. I Pods CoreDNS possono essere implementati sui nodi Fargate se il cluster include un AWS Fargate profilo con uno spazio dei nomi che corrisponde allo spazio per `deployment` CoreDNS. Per ulteriori informazioni su CoreDNS, consulta Utilizzo di CoreDNS per l'individuazione dei servizi nella documentazione Kubernetes.

La tabella seguente elenca la versione più recente del componente aggiuntivo di Amazon EKS disponibile per ogni versione Kubernetes.

Versione Kubernetes	**1.30**	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
	v1.11.1-eksbuild	v1.11.1-eksbuild	v1.10.1-eksbuild.1	v1.10.1-eksbuild.1	v1.9.3-eksbuild.	v1.9.3-eksbuild.	v1.9.3-eksbuild.	v1.8.7-eksbuild.10

> **Important**
>
> Se gestisci autonomamente questo componente aggiuntivo, le versioni nella tabella potrebbero non essere le stesse delle versioni autogestite disponibili. Per ulteriori

informazioni sull'aggiornamento del componente aggiuntivo del tipo autogestito, consulta Aggiornamento del componente aggiuntivo autogestito.

Considerazioni importanti sull'aggiornamento di CoreDNS

- Per migliorare la stabilità e la disponibilità di CoreDNS Deployment, le versioni `v1.9.3-eksbuild.6` e successive e `v1.10.1-eksbuild.3` vengono distribuiti con un `PodDisruptionBudget`. Se hai implementato un `PodDisruptionBudget` esistente, l'aggiornamento a queste versioni potrebbe non riuscire. Se l'aggiornamento non riesce, il problema dovrebbe essere risolto eseguendo una delle seguenti operazioni:
  - Quando esegui l'aggiornamento del componente aggiuntivo Amazon EKS, scegli di sovrascrivere le impostazioni esistenti come opzione di risoluzione dei conflitti. Se hai effettuato altre impostazioni personalizzate su Deployment, assicurati di eseguire il backup delle impostazioni prima dell'aggiornamento in modo da poter riapplicare le altre impostazioni personalizzate dopo l'aggiornamento.
  - Rimuovi `PodDisruptionBudget` esistente e riprova a eseguire l'aggiornamento.
- Nelle versioni aggiuntive EKS `v1.9.3-eksbuild.3` e successive e `v1.10.1-eksbuild.6` e successive, il CoreDNS Deployment imposta il `readinessProbe` per utilizzare l'endpoint `/ready`. Questo endpoint è abilitato nel file di configurazione `Corefile` per CoreDNS.

  Se si utilizza un dispositivo personalizzato `Corefile`, è necessario aggiungere il plugin `ready` alla configurazione, in modo che l'endpoint `/ready` sia attivo in CoreDNS e possa essere utilizzato dalla sonda.
- Nelle versioni aggiuntive EKS `v1.9.3-eksbuild.7` e successive e `v1.10.1-eksbuild.4` e successive, è possibile modificare il. `PodDisruptionBudget` È possibile modificare il componente aggiuntivo e modificare queste impostazioni nelle configurazioni facoltative utilizzando i campi dell'esempio seguente. Questo esempio mostra l'impostazione predefinita `PodDisruptionBudget`.

```
{
    "podDisruptionBudget": {
        "enabled": true,
        "maxUnavailable": 1
        }
}
```

È possibile impostare `maxUnavailable` o `minAvailable`, ma non è possibile impostare entrambi sullo stesso `PodDisruptionBudget`. Per ulteriori informazioni su `PodDisruptionBudgets`, consulta Specifica di un `PodDisruptionBudget` nella documentazione di Kubernetes.

Tieni presente che se imposti `enabled` su `false`, il `PodDisruptionBudget` non viene rimosso. Dopo aver impostato questo campo su `false`, è necessario eliminare l'oggetto `PodDisruptionBudget`. Allo stesso modo, se modifichi il componente aggiuntivo per utilizzarne una versione precedente (esegui il downgrade del componente aggiuntivo) dopo l'aggiornamento a una versione con un `PodDisruptionBudget`, questo non viene rimosso `PodDisruptionBudget`. Esegui il seguente comando per eliminare il `PodDisruptionBudget`:

```
kubectl delete poddisruptionbudget coredns -n kube-system
```

- Nelle versioni aggiuntive EKS `v1.10.1-eksbuild.5` e successive, modifica la tolleranza predefinita da `node-role.kubernetes.io/master:NoSchedule` a per conformarla `node-role.kubernetes.io/control-plane:NoSchedule` a KEP 2067. Per ulteriori informazioni su KEP 2067, consulta KEP-2067: rinominare l'etichetta "master" di kubeadm e la taint nelle KEP (Kubernetes Enhancement Proposal) su GitHub.

  Nelle versioni aggiuntive EKS `v1.8.7-eksbuild.8` e successive `v1.9.3-eksbuild.9` e successive, entrambe le tolleranze sono impostate per essere compatibili con tutte le versioni. Kubernetes
- Nelle versioni aggiuntive EKS `v1.9.3-eksbuild.11` `v1.10.1-eksbuild.7` e successive, CoreDNS Deployment imposta un valore predefinito per. `topologySpreadConstraints` Il valore predefinito garantisce che CoreDNS Pods siano distribuiti tra le zone di disponibilità se sono disponibili nodi in più zone di disponibilità. È possibile impostare un valore personalizzato che verrà utilizzato al posto del valore predefinito. Il valore predefinito è il seguente:

```
topologySpreadConstraints:
  - maxSkew: 1
    topologyKey: topology.kubernetes.io/zone
    whenUnsatisfiable: ScheduleAnyway
    labelSelector:
      matchLabels:
        k8s-app: kube-dns
```

CoreDNSv considerazioni sull'**1.11**aggiornamento

- Nelle versioni aggiuntive EKS `v1.11.1-eksbuild.4` e successive, l'immagine del contenitore si basa su un'immagine di base minima gestita da Amazon EKS Distro, che contiene un numero minimo di pacchetti e non dispone di shell. Per ulteriori informazioni, consultare Amazon EKS Distro. L'utilizzo e la risoluzione dei problemi dell'CoreDNSimmagine rimangono gli stessi.

Creare il componente aggiuntivo di Amazon EKS

Crea il componente aggiuntivo del tipo Amazon EKS. Verifica

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.

1. Scopri qual è la versione del componente aggiuntivo attualmente installata sul cluster.

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -d : -f 3
   ```

   Di seguito viene riportato un output di esempio:

   ```
   v1.10.1-eksbuild.11
   ```

2. Scopri qual è il tipo di componente aggiuntivo attualmente installato sul cluster. A seconda dello strumento con cui hai creato il cluster, al momento potresti non avere il componente aggiuntivo del tipo Amazon EKS installato sul cluster. Sostituisci *my-cluster* con il nome del cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query addon.addonVersion --output text
   ```

   Se viene restituito il numero di versione, sul cluster è installato il componente aggiuntivo del tipo Amazon EKS e non è necessario completare i passaggi rimanenti di questa procedura. Se viene restituito un errore, sul cluster non è installato il componente aggiuntivo del tipo Amazon EKS. Completa i passaggi rimanenti di questa procedura per installarlo.

3. Salva la configurazione del componente aggiuntivo attualmente installato.

```
kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
```

4. Crea il componente aggiuntivo utilizzando la AWS CLI. Se desideri utilizzare AWS Management Console o `eksctl` creare il componente aggiuntivo, consulta Creazione di un componente aggiuntivo e specifica `coredns` il nome del componente aggiuntivo. Copia il comando seguente sul tuo dispositivo. Apporta le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato.

   - Sostituisci *`my-cluster`* con il nome del cluster.
   - Sostituisci *`v1.11.1-eksbuild.9`* con la versione più recente elencata nella tabella delle versioni più recenti per la versione del cluster.

   ```
   aws eks create-addon --cluster-name my-cluster --addon-name coredns --addon-
   version v1.11.1-eksbuild.9
   ```

   Se hai applicato impostazioni personalizzate al tuo attuale componente aggiuntivo che sono in conflitto con le impostazioni predefinite del componente aggiuntivo di Amazon EKS, la creazione potrebbe non riuscire. Se la creazione non riesce, riceverai un errore che può aiutare a risolvere il problema. In alternativa, puoi aggiungere **`--resolve-conflicts OVERWRITE`** al comando precedente. Ciò consente al componente aggiuntivo di sovrascrivere le impostazioni personalizzate esistenti. Una volta creato il componente aggiuntivo, puoi aggiornarlo con le tue impostazioni personalizzate.

5. Verifica che la versione più recente del componente aggiuntivo per la versione Kubernetes del cluster sia stata aggiunta al cluster. Sostituisci *`my-cluster`* con il nome del cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   La creazione del componente aggiuntivo potrebbe richiedere alcuni secondi.

   Di seguito viene riportato un output di esempio:

   ```
   v1.11.1-eksbuild.9
   ```

6. Se hai creato impostazioni personalizzate per il componente aggiuntivo originale, prima di creare il componente aggiuntivo di Amazon EKS, utilizza la configurazione che hai salvato

precedentemente per aggiornare il componente aggiuntivo di Amazon EKS con le tue impostazioni personalizzate.

Aggiornamento del componente aggiuntivo di Amazon EKS

Aggiorna il componente aggiuntivo del tipo Amazon EKS. Se non hai aggiunto il componente aggiuntivo del tipo Amazon EKS al tuo cluster, aggiungilo o consulta Aggiornamento del componente aggiuntivo autogestito, invece di completare questa procedura.

1. Scopri qual è la versione del componente aggiuntivo attualmente installata sul cluster. Sostituisci *my-cluster* con il nome del cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    "addon.addonVersion" --output text
   ```

   Di seguito viene riportato un output di esempio:

   ```
   v1.10.1-eksbuild.11
   ```

   Se la versione restituita è la stessa della versione Kubernetes del cluster nella tabella delle versioni più recenti, la versione più recente è già installata nel cluster e non è necessario completare il resto di questa procedura. Se invece di un numero di versione ricevi un errore, il componente aggiuntivo del tipo Amazon EKS non è installato sul cluster. È necessario creare il componente aggiuntivo prima di poterlo aggiornare con questa procedura.
2. Salva la configurazione del componente aggiuntivo attualmente installato.

   ```
   kubectl get deployment coredns -n kube-system -o yaml > aws-k8s-coredns-old.yaml
   ```
3. Aggiorna il componente aggiuntivo utilizzando la AWS CLI. Se desideri utilizzare o aggiornare il componente aggiuntivo, consulta AWS Management Console . `eksctl` Aggiornamento di un componente aggiuntivo Copia il comando seguente sul tuo dispositivo. Apporta le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato.

   - Sostituisci *my-cluster* con il nome del cluster.
   - Sostituisci *v1.11.1-eksbuild.9* con la versione più recente elencata nella tabella delle versioni più recenti per la versione del cluster.

- L'opzione **`--resolve-conflicts`** *`CONSERVA`* mantiene i valori di configurazione esistenti per il componente aggiuntivo. Se hai impostato valori personalizzati per le impostazioni del componente aggiuntivo e non utilizzi questa opzione, Amazon EKS sovrascrive i tuoi valori con quelli predefiniti. Se utilizzi questa opzione, è preferibile testare eventuali modifiche ai campi e ai valori su un cluster non di produzione prima di aggiornare il componente aggiuntivo sul cluster di produzione. Se modifichi questo valore in `OVERWRITE`, tutte le impostazioni vengono modificate nei valori predefiniti di Amazon EKS. Se hai impostato valori personalizzati per un'impostazione qualunque, è possibile che vengano sovrascritti con i valori predefiniti di Amazon EKS. Se modifichi questo valore in `none`, Amazon EKS non modifica il valore di alcuna impostazione, ma l'aggiornamento potrebbe non riuscire. Se l'aggiornamento non riesce, riceverai un messaggio di errore che ti aiuterà a risolvere il conflitto.
- Se non stai aggiornando un'impostazione di configurazione, rimuovi **`--configuration-values '{`***`"replicaCount":3`***`}'`** dal comando. Se stai aggiornando un'impostazione di configurazione, sostituisci *`"replicaCount":3`* con l'impostazione desiderata. In questo esempio, il numero di repliche diCoreDNS è impostato su `3`. Il valore specificato deve essere valido per lo schema di configurazione. Se non conosci lo schema di configurazione, **`aws eks describe-addon-configuration --addon-name coredns --addon-version`** ***`v1.11.1-eksbuild.9`*** esegui sostituendo *`v1.11.1-eksbuild.9`* con il numero di versione del componente aggiuntivo di cui vuoi vedere la configurazione. Lo schema viene restituito nell'output. Se disponi di una configurazione personalizzata, desideri rimuoverla e reimpostare i valori di tutte le impostazioni ai valori predefiniti di Amazon EKS, rimuovi *`"replicaCount":3`* dal comando in modo da avere delle **`{}`** vuote. Per ulteriori informazioni sulle impostazioni CoreDNS, consulta Personalizzazione del servizio DNS nella documentazione di Kubernetes.

```
aws eks update-addon --cluster-name my-cluster --addon-name coredns --addon-
version v1.11.1-eksbuild.9 \
    --resolve-conflicts PRESERVE --configuration-values '{"replicaCount":3}'
```

Il completamento dell'aggiornamento potrebbe richiedere alcuni secondi.

4. Conferma che la versione del componente aggiuntivo sia stata aggiornata. Sostituisci *`my-cluster`* con il nome del cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns
```

Il completamento dell'aggiornamento potrebbe richiedere alcuni secondi.

Di seguito viene riportato un output di esempio:

```
{
    "addon": {
        "addonName": "coredns",
        "clusterName": "my-cluster",
        "status": "ACTIVE",
        "addonVersion": "v1.11.1-eksbuild.9",
        "health": {
            "issues": []
        },
        "addonArn": "arn:aws:eks:region:111122223333:addon/my-cluster/coredns/
d2c34f06-1111-2222-1eb0-24f64ce37fa4",
        "createdAt": "2023-03-01T16:41:32.442000+00:00",
        "modifiedAt": "2023-03-01T18:16:54.332000+00:00",
        "tags": {},
        "configurationValues": "{\"replicaCount\":3}"
    }
}
```

Aggiornamento del componente aggiuntivo autogestito

> **Important**
>
> Consigliamo di aggiungere al cluster il componente aggiuntivo del tipo Amazon EKS anziché quello del tipo autogestito. Se non conosci bene le differenze tra i due tipi, consulta la pagina the section called “Componenti aggiuntivi di Amazon EKS”. Per ulteriori informazioni sull'aggiunta di un componente aggiuntivo di Amazon EKS al cluster, consulta the section called “Creazione di un componente aggiuntivo”. Se non riesci a utilizzare il componente aggiuntivo Amazon EKS, ti consigliamo di segnalare un problema sul motivo per cui non puoi farlo all'archivio della roadmap GitHub di Containers.

1. Verifica che sul cluster sia installato il componente aggiuntivo del tipo autogestito. Sostituisci *my-cluster* con il nome del tuo cluster.

```
aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
 addon.addonVersion --output text
```

Se viene restituito un messaggio di errore, sul cluster è installato il componente aggiuntivo del tipo autogestito. Completa i passaggi rimanenti di questa procedura. Se viene restituito il numero di versione, sul cluster è installato il tipo Amazon EKS del componente aggiuntivo. Per aggiornare il tipo di Amazon EKS del componente aggiuntivo, utilizza la procedura descritta in Aggiornamento del componente aggiuntivo di Amazon EKS anziché questa. Se non conosci bene le differenze tra i due tipi di componente aggiuntivo, consulta Componenti aggiuntivi Amazon EKS.

2. Scopri qual è la versione dell'immagine di container attualmente installata sul cluster.

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

Di seguito viene riportato un output di esempio:

```
v1.8.7-eksbuild.2
```

3. Se la versione corrente di CoreDNS è `v1.5.0` o successiva, ma precedente alla versione elencata nella tabella Versioni CoreDNS, ignorare questa fase. Se la versione corrente è precedente alla `1.5.0`, è necessario modificare `ConfigMap` per CoreDNS al fine di utilizzare il componente aggiuntivo avanzato, anziché il componente aggiuntivo proxy.

   1. Aprire il file tramite il comando seguente:

      ```
      kubectl edit configmap coredns -n kube-system
      ```

   2. Sostituire `proxy` nella linea seguente con `forward`. Salvare il file e uscire dall'editor.

      ```
      proxy . /etc/resolv.conf
      ```

4. Se il cluster in origine è stato implementato su Kubernetes `1.17` o versioni precedenti, potrebbe essere necessario rimuovere una riga obsoleta dal manifesto CoreDNS.

> **Important**
>
> Devi completare questo passaggio prima di eseguire l'aggiornamento alla versione `1.7.0` di CoreDNS, ma è preferibile completare questo passaggio anche se esegui l'aggiornamento a una versione precedente.

1. Verificare se il manifesto CoreDNS presenta la riga di inserimento.

   ```
   kubectl get configmap coredns -n kube-system -o jsonpath='{$.data.Corefile}' |
    grep upstream
   ```

   Se non viene restituito alcun output, il manifesto non ha la riga di inserimento ed è possibile passare direttamente alla fase successiva per aggiornare CoreDNS. Se viene restituito l'output, è necessario rimuovere la riga di inserimento.

2. Modificare `ConfigMap` con il seguente comando, rimuovendo la riga nel file che ha la parola `upstream` in esso. Non apportare ulteriori modifiche al file. Una volta rimossa la riga di inserimento, salvare le modifiche.

   ```
   kubectl edit configmap coredns -n kube-system -o yaml
   ```

5. Recupera la versione dell'immagine CoreDNS corrente:

   ```
   kubectl describe deployment coredns -n kube-system | grep Image
   ```

   Di seguito viene riportato un output di esempio:

   ```
   602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.8.7-eksbuild.2
   ```

6. Se stai eseguendo l'aggiornamento a CoreDNS `1.8.3` o successive, allora dovrai aggiungere l'autorizzazione `endpointslices` al `clusterrole` Kubernetes `system:coredns`.

   ```
   kubectl edit clusterrole system:coredns -n kube-system
   ```

   Aggiungere le seguenti righe sotto le righe di autorizzazioni esistenti nella sezione `rules` del file.

   ```
   [...]
   - apiGroups:
     - discovery.k8s.io
     resources:
     - endpointslices
     verbs:
     - list
     - watch
   ```

```
[...]
```

7. Aggiorna il componente aggiuntivo CoreDNS sostituendo *602401143452* e *region-code* con i valori dell'output restituiti in un passaggio precedente. Sostituisci *v1.11.1-eksbuild.9* con la versione CoreDNSriportata nella tabella delle ultime versioni per la versione del Kubernetes.

```
kubectl set image deployment.apps/coredns -n kube-system
 coredns=602401143452.dkr.ecr.region-code.amazonaws.com/eks/coredns:v1.11.1-
eksbuild.9
```

   Di seguito viene riportato un output di esempio:

```
deployment.apps/coredns image updated
```

8. Controlla nuovamente la versione dell'immagine del container per accertarti che sia stata aggiornata alla versione specificata nel passaggio precedente.

```
kubectl describe deployment coredns -n kube-system | grep Image | cut -d ":" -f 3
```

   Di seguito viene riportato un output di esempio:

```
v1.11.1-eksbuild.9
```

Scalabilità automatica CoreDNS

Quando avvii un cluster Amazon EKS con almeno un Deployment nodo, per impostazione predefinita vengono distribuite due repliche dell'CoreDNSimmagine, indipendentemente dal numero di nodi distribuiti nel cluster. I CoreDNS Pod forniscono la risoluzione dei nomi per tutti i Pod del cluster. Le applicazioni utilizzano la risoluzione dei nomi per connettersi ai pod e ai servizi del cluster, nonché per connettersi ai servizi esterni al cluster. Con l'aumentare del numero di richieste di risoluzione dei nomi (interrogazioni) da parte dei pod, i CoreDNS pod possono diventare sovraccarichi e rallentare e rifiutare le richieste che i pod non sono in grado di gestire.

Per gestire l'aumento del carico sui CoreDNS pod, prendi in considerazione un sistema di scalabilità automatica per. CoreDNS Amazon EKS può gestire la scalabilità automatica della CoreDNS distribuzione nella versione aggiuntiva EKS di. CoreDNS Questo CoreDNS autoscaler monitora continuamente lo stato del cluster, incluso il numero di nodi e core della CPU. Sulla base di tali informazioni, il controller adatterà dinamicamente il numero di repliche dell'implementazione in

un cluster EKS. CoreDNS Questa funzionalità è disponibile per tutte le versioni di rilascio di EKS CoreDNS `v1.9` `1.25` e successive. Per ulteriori informazioni sulle versioni compatibili con CoreDNS Autoscaling, consulta la sezione seguente.

Si consiglia di utilizzare questa funzionalità insieme ad altre best practice di EKS Cluster Autoscaling per migliorare la disponibilità complessiva delle applicazioni e la scalabilità del cluster.

Prerequisiti

Affinché Amazon EKS possa scalare la tua CoreDNS distribuzione, sono necessari tre prerequisiti:

- È necessario utilizzare la versione aggiuntiva EKS di. CoreDNS
- Il cluster deve eseguire almeno le versioni minime del cluster e delle versioni della piattaforma.
- Il cluster deve eseguire almeno la versione minima del componente aggiuntivo EKS diCoreDNS.

Versione minima del cluster

La scalabilità automatica di CoreDNS viene eseguita da un nuovo componente nel piano di controllo del cluster, gestito da Amazon EKS. Per questo motivo, è necessario aggiornare il cluster a una versione EKS che supporti la versione minima della piattaforma che include il nuovo componente.

Un nuovo cluster Amazon EKS. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS. Il cluster deve essere Kubernetes versione `1.25` o successiva. Il cluster deve eseguire una delle Kubernetes versioni e delle versioni della piattaforma elencate nella tabella seguente o una versione successiva. Sono supportate anche tutte le versioni di Kubernetes e della piattaforma successive a quelle elencate. È possibile verificare la versione corrente di Kubernetes sostituendo *`my-cluster`* nel seguente comando con il nome del cluster e quindi eseguendo il comando modificato:

```
aws eks describe-cluster
              --name my-cluster --query cluster.version --output
              text
```

Versione Kubernetes	Versione della piattaforma
`1.29.3`	`eks.7`
`1.28.8`	`eks.13`
`1.27.12`	`eks.17`

Versione Kubernetes	Versione della piattaforma
1.26.15	eks.18
1.25.16	eks.19

> **Note**
>
> Sono supportate anche tutte Kubernetes le versioni successive della piattaforma, ad esempio 1.30 le Kubernetes eks.1 versioni successive.

Versione minima del componente aggiuntivo EKS

Versione Kubernetes	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**
	v1.11.1-eksbuild.9	v1.10.1-eksbuild.11	v1.10.1-eksbuild.11	v1.9.3-eksbuild.15	v1.9.3-eksbuild.15

Configurazione della scalabilità CoreDNS automatica in AWS Management Console

1. Assicurati che la versione del cluster sia uguale o superiore alla versione minima del cluster.

   Amazon EKS aggiorna automaticamente i cluster tra le versioni della stessa Kubernetes versione della piattaforma e non puoi avviare questo processo da solo. Puoi invece aggiornare il cluster alla Kubernetes versione successiva e il cluster verrà aggiornato alla versione K8s e alla versione più recente della piattaforma. Ad esempio, se esegui l'aggiornamento da 1.25 a1.26, il cluster verrà aggiornato a. 1.26.15 eks.18

   Le nuove versioni di Kubernetes hanno introdotto modifiche significative. Pertanto, si consiglia di testare il comportamento delle applicazioni utilizzando un cluster separato della nuova Kubernetes versione prima di aggiornare i cluster di produzione.

   Per aggiornare un cluster a una nuova Kubernetes versione, segui la procedura riportata in[Aggiornamento della versione di Kubernetes del cluster Amazon EKS](#).

2. Assicurati di avere il componente aggiuntivo EKS perCoreDNS, non la distribuzione autogestitaCoreDNS.

   A seconda dello strumento con cui hai creato il cluster, al momento potresti non avere il componente aggiuntivo del tipo Amazon EKS installato sul cluster. Per vedere quale tipo di componente aggiuntivo è installato nel cluster, è possibile eseguire il comando seguente. Sostituisci `my-cluster` con il nome del cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Se viene restituito un numero di versione, nel cluster è installato il tipo di componente aggiuntivo Amazon EKS e puoi continuare con il passaggio successivo. Se viene restituito un errore, sul cluster non è installato il componente aggiuntivo del tipo Amazon EKS. Completa i passaggi rimanenti della procedura Creare il componente aggiuntivo di Amazon EKS per sostituire la versione autogestita con il componente aggiuntivo Amazon EKS.

3. Assicurati che il tuo componente aggiuntivo CoreDNS EKS per abbia una versione uguale o superiore alla versione minima del componente aggiuntivo EKS.

   Scopri qual è la versione del componente aggiuntivo attualmente installata sul cluster. Puoi effettuare il check-in AWS Management Console o eseguire il seguente comando:

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   Di seguito viene riportato un output di esempio:

   ```
   v1.10.1-eksbuild.11
   ```

   Confrontate questa versione con la versione minima del componente aggiuntivo EKS nella sezione precedente. Se necessario, aggiorna il componente aggiuntivo EKS a una versione superiore seguendo la procedura. Aggiornamento del componente aggiuntivo di Amazon EKS

4. Aggiungete la configurazione di scalabilità automatica alle impostazioni di configurazione opzionali del componente aggiuntivo EKS.

a. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.

b. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui configurare il componente aggiuntivo.

c. Seleziona la scheda Componenti aggiuntivi.

d. Seleziona la casella in alto a destra del riquadro del CoreDNS componente aggiuntivo, quindi scegli Modifica.

e. Nella CoreDNS pagina Configura:

i. seleziona la Version (Versione) da utilizzare. Ti consigliamo di mantenere la stessa versione del passaggio precedente e di aggiornare la versione e la configurazione con azioni separate.

ii. Scegli Impostazioni di configurazione facoltative.

iii. Inserisci la chiave JSON **`"autoscaling":`** e il valore di un oggetto JSON annidato con una chiave **`"enabled":`** e un valore **`true`** in Valori di configurazione. Il testo risultante deve essere un oggetto JSON valido. Se questa chiave e questo valore sono gli unici dati nella casella di testo, racchiudi la chiave e il valore tra parentesi graffe `{}`. L'esempio seguente mostra che la scalabilità automatica è abilitata:

```
{
  "autoScaling": {
    "enabled": true
  }
}
```

iv. (Facoltativo) È possibile fornire valori minimi e massimi a cui la scalabilità automatica può scalare il numero di pod. CoreDNS

L'esempio seguente mostra che la scalabilità automatica è abilitata e tutte le chiavi opzionali hanno dei valori. Si consiglia che il numero minimo di CoreDNS pod sia sempre maggiore di 2 per garantire la resilienza del servizio DNS nel cluster.

```
{
  "autoScaling": {
    "enabled": true,
    "minReplicas": 2,
    "maxReplicas": 10
```

```
  }
}
```

f. Per applicare la nuova configurazione sostituendo i CoreDNS pod, scegli Salva modifiche.

Amazon EKS applica le modifiche ai componenti aggiuntivi EKS utilizzando un'implementazione di Kubernetes Deployment for CoredNS. Puoi tenere traccia dello stato dell'implementazione nella cronologia degli aggiornamenti del componente aggiuntivo in e con. AWS Management Console `kubectl rollout status deployment/coredns --namespace kube-system`

`kubectl rolloutha` i seguenti comandi:

```
$ kubectl rollout

history  -- View rollout history
pause    -- Mark the provided resource as paused
restart  -- Restart a resource
resume   -- Resume a paused resource
status   -- Show the status of the rollout
undo     -- Undo a previous rollout
```

Se l'implementazione richiede troppo tempo, Amazon EKS annullerà l'implementazione e un messaggio con il tipo di Addon Update e lo stato Failed verrà aggiunto alla cronologia degli aggiornamenti del componente aggiuntivo. Per indagare su eventuali problemi, inizia dalla cronologia dell'implementazione ed eseguilo `kubectl logs` su un CoreDNS pod per visualizzare i log di. CoreDNS

5. Se la nuova voce nella cronologia degli aggiornamenti ha lo stato Riuscito, l'implementazione è stata completata e il componente aggiuntivo utilizza la nuova configurazione in tutti i pod. CoreDNS Man mano che modifichi il numero di nodi e core di CPU dei nodi nel cluster, Amazon EKS ridimensiona il numero di repliche della distribuzione. CoreDNS

Configurazione della scalabilità automatica CoreDNS in AWS Command Line Interface

1. Assicurati che la versione del cluster sia uguale o superiore alla versione minima del cluster.

Amazon EKS aggiorna automaticamente i cluster tra le versioni della stessa Kubernetes versione della piattaforma e non puoi avviare questo processo da solo. Puoi invece aggiornare il cluster

alla Kubernetes versione successiva e il cluster verrà aggiornato alla versione K8s e alla versione più recente della piattaforma. Ad esempio, se esegui l'aggiornamento da `1.25` a`1.26`, il cluster verrà aggiornato a. `1.26.15 eks.18`

Le nuove versioni di Kubernetes hanno introdotto modifiche significative. Pertanto, si consiglia di testare il comportamento delle applicazioni utilizzando un cluster separato della nuova Kubernetes versione prima di aggiornare i cluster di produzione.

Per aggiornare un cluster a una nuova Kubernetes versione, segui la procedura riportata in[Aggiornamento della versione di Kubernetes del cluster Amazon EKS](#).

2. Assicurati di avere il componente aggiuntivo EKS perCoreDNS, non la distribuzione autogestitaCoreDNS.

   A seconda dello strumento con cui hai creato il cluster, al momento potresti non avere il componente aggiuntivo del tipo Amazon EKS installato sul cluster. Per vedere quale tipo di componente aggiuntivo è installato nel cluster, è possibile eseguire il comando seguente. Sostituisci `my-cluster` con il nome del cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns --query
    addon.addonVersion --output text
   ```

   Se viene restituito il numero di versione, sul cluster è installato il tipo Amazon EKS del componente aggiuntivo. Se viene restituito un errore, sul cluster non è installato il componente aggiuntivo del tipo Amazon EKS. Completa i passaggi rimanenti della procedura [Creare il componente aggiuntivo di Amazon EKS](#) per sostituire la versione autogestita con il componente aggiuntivo Amazon EKS.

3. Assicurati che il tuo componente aggiuntivo CoreDNS EKS per abbia una versione uguale o superiore alla versione minima del componente aggiuntivo EKS.

   Scopri qual è la versione del componente aggiuntivo attualmente installata sul cluster. Puoi effettuare il check-in AWS Management Console o eseguire il seguente comando:

   ```
   kubectl describe deployment coredns --namespace kube-system | grep coredns: | cut -
   d : -f 3
   ```

   Di seguito viene riportato un output di esempio:

```
v1.10.1-eksbuild.11
```

Confrontate questa versione con la versione minima del componente aggiuntivo EKS nella sezione precedente. Se necessario, aggiorna il componente aggiuntivo EKS a una versione superiore seguendo la procedura. Aggiornamento del componente aggiuntivo di Amazon EKS

4. Aggiungete la configurazione di scalabilità automatica alle impostazioni di configurazione opzionali del componente aggiuntivo EKS.

   Eseguite il comando seguente. AWS CLI Sostituisci `my-cluster` con il nome del cluster e l'ARN del ruolo IAM con il ruolo che stai utilizzando.

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
       --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
   {"enabled":true}}'
   ```

   Amazon EKS applica le modifiche ai componenti aggiuntivi EKS utilizzando un'implementazione di Kubernetes Deployment for CoredNS. Puoi tenere traccia dello stato dell'implementazione nella cronologia degli aggiornamenti del componente aggiuntivo in e con. AWS Management Console `kubectl rollout status deployment/coredns --namespace kube-system`

   `kubectl rolloutha` i seguenti comandi:

   ```
   kubectl rollout

   history  -- View rollout history
   pause    -- Mark the provided resource as paused
   restart  -- Restart a resource
   resume   -- Resume a paused resource
   status   -- Show the status of the rollout
   undo     -- Undo a previous rollout
   ```

   Se l'implementazione richiede troppo tempo, Amazon EKS annullerà l'implementazione e un messaggio con il tipo di Addon Update e lo stato Failed verrà aggiunto alla cronologia degli aggiornamenti del componente aggiuntivo. Per indagare su eventuali problemi, inizia dalla cronologia dell'implementazione ed eseguilo `kubectl logs` su un CoreDNS pod per visualizzare i log di. CoreDNS

5. (Facoltativo) È possibile fornire valori minimi e massimi a cui la scalabilità automatica può scalare il numero di pod. CoreDNS

   L'esempio seguente mostra che la scalabilità automatica è abilitata e tutte le chiavi opzionali hanno dei valori. Si consiglia che il numero minimo di CoreDNS pod sia sempre maggiore di 2 per garantire la resilienza del servizio DNS nel cluster.

   ```
   aws eks update-addon --cluster-name my-cluster --addon-name coredns \
       --resolve-conflicts PRESERVE --configuration-values '{"autoScaling":
   {"enabled":true}, "minReplicas": 2, "maxReplicas": 10}'
   ```

6. Controlla lo stato dell'aggiornamento del componente aggiuntivo eseguendo il seguente comando:

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name coredns \
   ```

   Se vedi questa riga:`"status": "ACTIVE"`, significa che l'implementazione è stata completata e il componente aggiuntivo sta utilizzando la nuova configurazione in tutti i pod. CoreDNS Man mano che modifichi il numero di nodi e core di CPU dei nodi nel cluster, Amazon EKS ridimensiona il numero di repliche della distribuzione. CoreDNS

Parametri di CoreDNS

CoreDNS come componente aggiuntivo di EKS espone i parametri di CoreDNS sulla porta `9153` nel formato Prometheus nel servizio `kube-dns`. Puoi usare Prometheus, l'agente Amazon CloudWatch o qualsiasi altro sistema compatibile per eseguire lo scraping (raccogliere) questi parametri.

Per un esempio di configurazione di scraping compatibile sia con Prometheus che con l'agente CloudWatch, consulta Configurazione dell'agente CloudWatch per Prometheus nella Guida per l'utente di Amazon CloudWatch.

Utilizzo del componente aggiuntivo Kubernetes **`kube-proxy`**

> **Important**
>
> Consigliamo di aggiungere al cluster il componente aggiuntivo del tipo Amazon EKS anziché quello del tipo autogestito. Se non conosci bene le differenze tra i due tipi, consulta la

pagina the section called “Componenti aggiuntivi di Amazon EKS”. Per ulteriori informazioni sull'aggiunta di un componente aggiuntivo di Amazon EKS al cluster, consulta the section called “Creazione di un componente aggiuntivo”. Se non riesci a utilizzare il componente aggiuntivo Amazon EKS, ti consigliamo di segnalare un problema sul motivo per cui non puoi farlo all'archivio della roadmap GitHub di Containers.

Il componente aggiuntivo `kube-proxy` viene implementato su ogni nodo Amazon EC2 del cluster Amazon EKS. Mantiene le regole di rete sui nodi e consente la comunicazione di rete con i tuoi Pods. Il componente aggiuntivo non viene implementato sui nodi Fargate nel cluster. Per ulteriori informazioni, consulta `kube-proxy` nella documentazione Kubernetes.

La tabella seguente elenca la versione più recente del componente aggiuntivo di Amazon EKS disponibile per ogni versione Kubernetes.

Versione Kubernetes	**1.30**	**1.29**	**1.28**	**1.27**	**1.26**	**1.25**	**1.24**	**1.23**
	v1.30.0-eksbuild.3	v1.29.3-eksbuild.5	v1.28.8-eksbuild.5	v1.27.16-eksbuild.14	v1.26.15-eksbuild.18	v1.25.16-eksbuild.18	v1.24.17-eksbuild.18	v1.23.17-eksbuild.19

> **Important**
>
> Una versione precedente della documentazione non era corretta. `kube-proxy`versioni `v1.28.5`v1.27.9, e `v1.26.12` non sono disponibili.
> Se gestisci autonomamente questo componente aggiuntivo, le versioni nella tabella potrebbero non essere le stesse delle versioni autogestite disponibili.

Esistono due tipi di immagine di container `kube-proxy` disponibili per ogni versione del cluster Amazon EKS:

- Default (Predefinito): questo tipo è basato su un'immagine Docker basata su Debian mantenuta dalla comunità upstream Kubernetes.

- Minimal (Minimo): questo tipo è basato su un'immagine di base minima mantenuta da Amazon EKS Distro che contiene pacchetti minimi senza shell (interprete di comandi). Per ulteriori informazioni, consultare Amazon EKS Distro.

Ultima versione dell'immagine di container **`kube-proxy`** autogestita disponibile per ogni versione del cluster Amazon EKS

Image type (Tipo di immagine)	**`1.30`**	**`1.29`**	**`1.28`**	**`1.27`**	**`1.26`**	**`1.25`**	**`1.24`**	**`1.23`**
`kube-proxy` (tipo predefinito)	È disponibile solo il tipo minimo	È disponibile solo il tipo minimo	È disponibile solo il tipo minimo	È disponibile solo il tipo minimo	È disponibile solo il tipo minimo	È disponibile solo il tipo minimo	`v1.24.1-eksbuild.2`	`v1.23.16-eksbuild.2`
`kube-proxy` (tipo minimo)	`v1.30.0-minimal-eksbuild.`	`v1.29.3-minimal-eksbuild.`	`v1.28.8-minimal-eksbuild.`	`v1.27.1-minimal-eksbuild`	`v1.26.1-minimal-eksbuild`	`v1.25.1-minimal-eksbuild`	`v1.24.1-minimal-eksbuild`	`v1.23.17-minimal-eksbuild.5`

Important

- Il tipo di immagine predefinito non è disponibile per Kubernetes versione `1.25` e successive. È necessario utilizzare il tipo di immagine minimo.
- Quando aggiorni un tipo di componente aggiuntivo Amazon EKS, specifichi una versione del componente aggiuntivo Amazon EKS valida, che potrebbe non essere una versione elencata in questa tabella. Questo accade perché le versioni dei componenti aggiuntivi di Amazon EKS non sempre corrispondono alle versioni delle immagini del container specificate durante l'aggiornamento del componente aggiuntivo del tipo autogestito. Quando si aggiorna il componente aggiuntivo del tipo autogestito, si specifica una versione valida dell'immagine di container indicata in questa tabella.

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.

Considerazioni

- `Kube-proxy` su un cluster Amazon EKS ha la stessa policy di compatibilità e disallineamento di Kubernetes. Scopri come effettuare il Recupera la compatibilità delle versioni del componente aggiuntivo.
- `Kube-proxy` deve essere alla stessa versione secondaria di `kubelet` sui nodi Amazon EC2.
- `Kube-proxy` non può essere successivo alla versione secondaria del piano di controllo (control plane) del tuo cluster.
- Se hai aggiornato recentemente il cluster a una nuova versione secondaria di Kubernetes, aggiorna i nodi EC2 Amazon alla medesima versione secondaria prima di eseguire l'aggiornamento di `kube-proxy` alla stessa versione secondaria dei nodi.

Aggiornamento del componente aggiuntivo autogestito **`kube-proxy`**

1. Verifica che sul cluster sia installato il componente aggiuntivo del tipo autogestito. Sostituisci *`my-cluster`* con il nome del tuo cluster.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name kube-proxy --query
    addon.addonVersion --output text
   ```

   Se viene restituito un messaggio di errore, sul cluster è installato il componente aggiuntivo del tipo autogestito. I passaggi rimanenti di questo argomento riguardano l'aggiornamento del componente aggiuntivo del tipo autogestito. Se viene restituito il numero di versione, sul cluster è installato il componente aggiuntivo del tipo Amazon EKS. Per aggiornarlo, utilizza la procedura descritta nell'argomento Aggiornamento di un componente aggiuntivo anziché quella descritta in questo argomento. Se non conosci bene le differenze tra i due tipi di componente aggiuntivo, consulta la pagina Componenti aggiuntivi Amazon EKS.
2. Scopri qual è la versione dell'immagine di container attualmente installata sul cluster.

   ```
   kubectl describe daemonset kube-proxy -n kube-system | grep Image
   ```

Di seguito viene riportato un output di esempio:

```
Image:    602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.29.1-eksbuild.2
```

Nell'output di esempio, *v1.29.1-eksbuild.2* è la versione installata nel cluster.

3. Aggiorna il componente aggiuntivo kube-proxy sostituendo *602401143452* e *region-code* con i valori del tuo output. Nel passaggio precedente, sostituisci *v1.30.0-eksbuild.3* con la versione di kube-proxy elencata nella tabella Ultima versione di immagine di container di kube-proxy disponibile per ogni versione del cluster Amazon EKS. Puoi specificare il numero di versione per il tipo di immagine predefinito o minimo.

```
kubectl set image daemonset.apps/kube-proxy -n kube-system kube-proxy=602401143452.dkr.ecr.region-code.amazonaws.com/eks/kube-proxy:v1.30.0-eksbuild.3
```

Di seguito viene riportato un output di esempio:

```
daemonset.apps/kube-proxy image updated
```

4. Verifica che la nuova versione sia ora installata nel cluster.

```
kubectl describe daemonset kube-proxy -n kube-system | grep Image | cut -d ":" -f 3
```

Di seguito viene riportato un output di esempio:

```
v1.30.0-eksbuild.3
```

5. Se utilizzi nodi x86 e Arm nello stesso cluster e il cluster è stato implementato il prima del 17 agosto 2020. Modificare, il manifesto kube-proxy attraverso il seguente comando, per includere un selettore di nodi per più architetture hardware. Questa è un'operazione da compiere una sola volta. Dopo aver aggiunto il selettore al manifesto, non occorrerà aggiungerlo ogni volta che aggiorni il componente aggiuntivo. Se il cluster è stato implementato in data 17 agosto 2020 o successiva, il kube-proxy è già in grado di supportare una multi-architettura.

```
kubectl edit -n kube-system daemonset/kube-proxy
```

Aggiungere il seguente selettore di nodo al file nell'editor, quindi salvare il file. Per un esempio di dove includere questo testo nell'editor, consulta File manifesto CNI su GitHub. Ciò consente a Kubernetes di estrarre l'immagine hardware corretta in base all'architettura hardware del nodo.

```
- key: "kubernetes.io/arch"
  operator: In
  values:
  - amd64
  - arm64
```

6. Se il cluster è stato originariamente creato con Kubernetes versione `1.14` o successiva, puoi saltare questo passaggio perché `kube-proxy` include già questo `Affinity Rule`. Se in origine hai creato un cluster Amazon EKS con la versione `1.13` di Kubernetes o precedente e intendi utilizzare nodi Fargate nel tuo cluster, modifica il manifesto `kube-proxy` in modo da includere una regola `NodeAffinity` per impedire ai Pods `kube-proxy` la pianificazione su nodi Fargate. Questa modifica si applica una sola volta. Dopo aver aggiunto `Affinity Rule` al manifesto, non occorrerà aggiungerlo ogni volta che aggiorni il componente aggiuntivo. Modifica il `DaemonSet` `kube-proxy`.

   ```
   kubectl edit -n kube-system daemonset/kube-proxy
   ```

   Aggiungi i seguenti elementi `Affinity Rule` alle sezioni `DaemonSet` `spec` del file nell'editor e salvare il file. Per un esempio di dove includere questo testo nell'editor, consulta File manifesto CNI su GitHub.

   ```
   - key: eks.amazonaws.com/compute-type
     operator: NotIn
     values:
     - fargate
   ```

Accedi all'Elastic Kubernetes Service di Amazon utilizzando un endpoint di interfaccia (AWS PrivateLink)

Puoi utilizzarlo AWS PrivateLink per creare una connessione privata tra il tuo VPC e Amazon Elastic Kubernetes Service. Puoi accedere ad Amazon EKS come se fosse nel tuo VPC, senza l'uso di

un gateway Internet, un dispositivo NAT, una connessione VPN o una connessione. AWS Direct Connect Le istanze nel tuo VPC non richiedono indirizzi IP pubblici per l'accesso ad Amazon EKS.

Stabilisci questa connessione privata creando un endpoint di interfaccia alimentato da AWS PrivateLink. In ciascuna sottorete viene creata un'interfaccia di rete endpoint da abilitare per l'endpoint di interfaccia. Queste sono interfacce di rete gestite dal richiedente che fungono da punto di ingresso per il traffico destinato ad Amazon EKS.

Per ulteriori informazioni, consulta la sezione Accesso a Servizi AWS tramite AWS PrivateLink nella Guida di AWS PrivateLink .

Considerazioni per Amazon EKS

- Prima di configurare un endpoint VPC di interfaccia per Amazon EKS, rivedi Considerations (Considerazioni) nella Guida di AWS PrivateLink .
- Amazon EKS supporta l'esecuzione di chiamate a tutte le azioni API all'interno dell'endpoint dell'interfaccia, ma non alle API Kubernetes. Il server API Kubernetes già supporta un endpoint privato. L'endpoint privato del server API Kubernetes crea un endpoint privato per il server API Kubernetes che utilizzi per comunicare con il tuo cluster (tramite strumenti di gestione Kubernetes, ad esempio `kubectl`). Puoi abilitare l'accesso privato al server Kubernetes API in modo che tutte le comunicazioni tra i tuoi nodi e il server API rimangano all'interno del tuo VPC. AWS PrivateLink per l'API Amazon EKS ti aiuta a chiamare le API di Amazon EKS dal tuo VPC senza esporre il traffico alla rete Internet pubblica.
- Non puoi configurare Amazon EKS in modo che sia accessibile solo tramite un endpoint di interfaccia.
- I prezzi standard AWS PrivateLink si applicano agli endpoint di interfaccia per Amazon EKS. La fatturazione avviene per ogni ora di provisioning di un endpoint di interfaccia in ogni zona di disponibilità e per i dati elaborati tramite l'endpoint di interfaccia. Per ulteriori informazioni, consultare Prezzi di AWS PrivateLink.
- Le policy degli endpoint VPC non sono supportate per Amazon EKS. Per impostazione predefinita, l'accesso completo ad Amazon EKS è consentito tramite l'endpoint di interfaccia. In alternativa, puoi associare un gruppo di sicurezza alle interfacce di rete degli endpoint per controllare il traffico verso Amazon EKS tramite l'endpoint di interfaccia.
- Puoi utilizzare i log di flusso del VPC per acquisire informazioni sul traffico IP da/verso le interfacce di rete, inclusi gli endpoint di interfaccia. Puoi pubblicare i dati del log di flusso su Amazon CloudWatch o Amazon S3. Per ulteriori informazioni, consulta Logging IP traffic using VPC Flow

Logs (Registrazione del traffico IP utilizzando log di flusso VPC) nella Guida per l'utente di Amazon VPC.

- Puoi accedere alle API Amazon EKS da un data center on-premise connettendolo a un VPC dotato di un endpoint di interfaccia. Puoi usare AWS Direct Connect o AWS Site-to-Site VPN connettere i tuoi siti locali a un VPC.
- Puoi connettere altri VPC al VPC con un endpoint di interfaccia utilizzando un peering VPC o AWS Transit Gateway . Il peering VPC è una connessione di rete tra due VPC. Puoi stabilire una connessione peering VPC tra i tuoi VPC oppure con un VPC in un altro account. I VPC possono essere in diversi formati. Regioni AWS Il traffico tra VPC peer rimane sulla rete. AWS Il traffico non attraversa la rete Internet pubblica. Un gateway di transito è un hub di transito di rete che puoi utilizzare per intercollegare VPC. Il traffico tra un VPC e un gateway di transito rimane sulla rete privata globale AWS . Il traffico non è esposto alla rete Internet pubblica.
- Gli endpoint di interfaccia VPC per Amazon EKS sono accessibili solo tramite `IPv4`. `IPv6` non è supportato.
- AWS PrivateLink il supporto non è disponibile in Asia Pacifico (Hyderabad), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Canada occidentale (Calgary), Europa (Spagna), Europa (Zurigo) o Medio Oriente (Emirati Arabi Uniti). Regioni AWS

Crea un endpoint di interfaccia per Amazon EKS

Puoi creare un endpoint di interfaccia per Amazon EKS utilizzando la console Amazon VPC o AWS Command Line Interface ().AWS CLI Per ulteriori informazioni, consulta la sezione Creazione di un endpoint VPC nella Guida di AWS PrivateLink .

Crea un endpoint di interfaccia per Amazon EKS utilizzando il seguente nome servizio:

```
com.amazonaws.region-code.eks
```

Per impostazione predefinita, la funzione DNS privato è abilitata quando viene creato un endpoint di interfaccia per Amazon EKS e altri Servizi AWS. Tuttavia, devi accertarti che i seguenti attributi VPC siano impostati su `true`: `enableDnsHostnames` e `enableDnsSupport`. Per ulteriori informazioni, consulta Visualizzazione e aggiornamento degli attributi DNS per il VPC nella Guida per l'utente di Amazon VPC. Con la funzione DNS privato abilitata per l'endpoint di interfaccia:

- Puoi effettuare qualunque richiesta API ad Amazon EKS utilizzando il nome DNS regionale predefinito. Ad esempio, `eks.region.amazonaws.com`. Per un elenco delle API, consulta Actions (Azioni) nella documentazione di riferimento dell'API Amazon EKS.

- Non è necessario apportare modifiche alle applicazioni che invocano le API EKS.
- Qualsiasi chiamata effettuata all'endpoint del servizio predefinito di Amazon EKS viene instradata automaticamente attraverso l'endpoint dell'interfaccia sulla rete privata. AWS

Carichi di lavoro

I carichi di lavoro vengono implementati in container, a loro volta implementati in Pods in Kubernetes. Un Pod include uno o più container. In genere, uno o più Pods che forniscono lo stesso servizio vengono implementati in un servizio Kubernetes. Dopo aver implementato più Pods che forniscono lo stesso servizio, è possibile:

- Visualizzare le informazioni sui carichi di lavoro in esecuzione su ogni cluster utilizzando il AWS Management Console.
- Aumenta o riduci i Pods pod con Kubernetes Vertical Pod Autoscaler.
- Aumenta o riduci il numero di Pods necessari per soddisfare la domanda con i Kubernetes Horizontal Pod Autoscaler.
- Crea un network load balancer esterno (per Pods accessibili a Internet) o interno (per Pods privati) to balance network traffic across Pods. Il load balancer indirizza il traffico al livello 4 del modello OSI.
- Crea un Bilanciamento del carico di applicazione su Amazon EKS per bilanciare il traffico delle applicazioni tra i Pods. Il load balancer di applicazione indirizza il traffico al livello 7 del modello OSI.
- Se sei un nuovo utente di Kubernetes, questo argomento aiuta a Implementare un'applicazione di esempio.
- È possibile limitare gli indirizzi IP assegnabili a un servizio con `externalIPs`.

Implementare un'applicazione di esempio

In questa sezione, imparerai a implementare un'applicazione di esempio sul cluster.

Prerequisiti

- Un cluster Kubernetes esistente cluster con almeno un nodo. Se non si dispone di un cluster Amazon EKS esistente, è possibile implementarne uno utilizzando una delle guide su Guida introduttiva ad Amazon EKS. Se stai eseguendo l'implementazione di un'applicazione Windows, allora il supporto Windows deve essere abilitato per il cluster e almeno un nodo Windowsdi Amazon EC2.
- `Kubectl` installato sul computer. Per ulteriori informazioni, consulta Installazione o aggiornamento di `kubectl`.

- `Kubectl` configurato per comunicare con il cluster. Per ulteriori informazioni, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.
- Se si prevede di implementare il carico di lavoro di esempio in Fargate, è necessario disporre di un profilo Fargate che include lo stesso spazio dei nomi creato in questo tutorial, che è `eks-sample-app`, a meno che non cambi il nome. Se hai usato una delle guide alle operazioni di base per creare il cluster, dovrai creare un nuovo profilo o aggiungere lo spazio dei nomi al profilo esistente, poiché il profilo creato nelle guide introduttive non specifica lo spazio dei nomi utilizzato in questo tutorial. Il VPC deve disporre di almeno una sottorete privata.

Per implementare un'applicazione di esempio

Sebbene molte variabili siano modificabili nei passaggi seguenti, si consiglia di modificare solo i valori delle variabili, se specificato. Con una migliore conoscenza dei Pods, delle implementazioni e dei servizi Kubernetes, puoi provare a modificare altri valori.

1. Crea uno spazio dei nomi . Uno spazio dei nomi consente di raggruppare le risorse in Kubernetes. Per ulteriori informazioni, consulta Spazi dei nomi nella documentazione di Kubernetes. Se si prevede di implementare l'applicazione di esempio su AWS Fargate, assicurati che il valore per `namespace` nel AWS Fargate profilo sia `eks-sample-app`.

   ```
   kubectl create namespace eks-sample-app
   ```

2. Crea una implementazione Kubernetes. Questa implementazione di esempio estrae un'immagine container da un repository pubblico e ne implementa tre repliche (singoli Pods) sul cluster. Per ulteriori informazioni, consulta Implementazioni nella documentazione di Kubernetes. Puoi implementare l'applicazione su nodi Linux o Windows. Se stai eseguendo l'implementazione su Fargate, potrai implementare solo un'applicazione Linux.

   a. Salva i seguenti contenuti in un file denominato `eks-sample-deployment.yaml`. I container nell'applicazione di esempio non utilizzano l'archiviazione di rete, ma potrebbe essere necessaria per determinate applicazioni. Per ulteriori informazioni, consulta Archiviazione.

      Linux

      `amd64` o `arm64` `values` nella chiave `kubernetes.io/arch` significa che l'applicazione può essere implementata su entrambe le architetture hardware (se entrambe sono presenti nel cluster). Ciò è possibile perché questa immagine

è un'immagine multi-architettura, ma non tutte lo sono. È possibile determinare l'architettura hardware su cui è supportata l'immagine visualizzando i dettagli dell'immagine nel repository da cui la stai estraendo. Quando si implementano immagini che non supportano un tipo di architettura hardware o che non si desidera implementare, rimuovere quel tipo ti architettura dal manifesto. Per ulteriori informazioni, consulta Etichette, annotazioni e taint note nella documentazione di Kubernetes.

`kubernetes.io/os: linux` `nodeSelector` significa che se avessi nodi Linux e Windows, ad esempio nel cluster, l'immagine verrebbe implementata solo sui nodi Linux. Per ulteriori informazioni, consulta Etichette, annotazioni e taint note nella documentazione di Kubernetes.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eks-sample-linux-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-linux-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: eks-sample-linux-app
  template:
    metadata:
      labels:
        app: eks-sample-linux-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: kubernetes.io/arch
                operator: In
                values:
                - amd64
                - arm64
      containers:
      - name: nginx
        image: public.ecr.aws/nginx/nginx:1.23
```

```
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
      nodeSelector:
        kubernetes.io/os: linux
```

Windows

`kubernetes.io/os: windows` `nodeSelector` significa che se avessi nodi Windows e Linux, ad esempio nel cluster, l'immagine verrebbe implementata solo sui nodi Windows. Per ulteriori informazioni, consulta Etichette, annotazioni e taint note nella documentazione di Kubernetes.

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eks-sample-windows-deployment
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  replicas: 3
  selector:
    matchLabels:
      app: eks-sample-windows-app
  template:
    metadata:
      labels:
        app: eks-sample-windows-app
    spec:
      affinity:
        nodeAffinity:
          requiredDuringSchedulingIgnoredDuringExecution:
            nodeSelectorTerms:
            - matchExpressions:
              - key: beta.kubernetes.io/arch
                operator: In
                values:
                - amd64
      containers:
      - name: windows-server-iis
        image: mcr.microsoft.com/windows/servercore:ltsc2019
```

```
        ports:
        - name: http
          containerPort: 80
        imagePullPolicy: IfNotPresent
        command:
        - powershell.exe
        - -command
        - "Add-WindowsFeature Web-Server; Invoke-WebRequest -UseBasicParsing
 -Uri 'https://dotnetbinaries.blob.core.windows.net/servicemonitor/2.0.1.6/
ServiceMonitor.exe' -OutFile 'C:\\ServiceMonitor.exe'; echo
 '<html><body><br/><br/><marquee><H1>Hello EKS!!!<H1><marquee></body><html>'
 > C:\\inetpub\\wwwroot\\default.html; C:\\ServiceMonitor.exe 'w3svc'; "
      nodeSelector:
        kubernetes.io/os: windows
```

b. Applicare il manifesto di implementazione al cluster.

```
kubectl apply -f eks-sample-deployment.yaml
```

3. Crea un servizio. Un servizio consente di accedere a tutte le repliche tramite un unico indirizzo IP o nome. Per ulteriori informazioni, consulta Servizi nella documentazione di Kubernetes. Sebbene non sia implementato nell'applicazione di esempio, se disponi di applicazioni che devono interagire con altri AWS servizi, ti consigliamo di creare account di Kubernetes servizio per i tuoi Pods account e associarli agli account AWS IAM. Specificando gli account di servizio, i Pods avranno solo le autorizzazioni minime necessarie per interagire con altri servizi. Per ulteriori informazioni, consulta Ruoli IAM per gli account di servizio.

   a. Salva il seguente contenuto in un file denominato `eks-sample-service.yaml`. Kubernetes assegna al servizio il proprio indirizzo IP a cui è possibile accedere solo dall'interno del cluster. Per accedere al servizio dall'esterno del cluster, implementare il AWS Load Balancer Controller per bilanciare il carico dell'applicazione o del traffico di rete verso il servizio.

      Linux

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-linux-service
  namespace: eks-sample-app
  labels:
```

```
    app: eks-sample-linux-app
spec:
  selector:
    app: eks-sample-linux-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

Windows

```
apiVersion: v1
kind: Service
metadata:
  name: eks-sample-windows-service
  namespace: eks-sample-app
  labels:
    app: eks-sample-windows-app
spec:
  selector:
    app: eks-sample-windows-app
  ports:
    - protocol: TCP
      port: 80
      targetPort: 80
```

b. Applicare il manifesto del servizio al cluster.

```
kubectl apply -f eks-sample-service.yaml
```

4. Visualizzare tutte le risorse nello spazio dei nomi `eks-sample-app`.

```
kubectl get all -n eks-sample-app
```

Di seguito viene riportato un output di esempio:

Se hai implementato risorse Windows, allora tutte le istanze di *`linux`* nel seguente output saranno `windows`. Gli altri *`valori di esempio`* possono essere diversi dall'output.

```
NAME                                                READY   STATUS    RESTARTS   AGE
pod/eks-sample-linux-deployment-65b7669776-m6qxz    1/1     Running   0          27m
pod/eks-sample-linux-deployment-65b7669776-mmxvd    1/1     Running   0          27m
```

```
pod/eks-sample-linux-deployment-65b7669776-qzn22   1/1     Running   0          27m

NAME                                TYPE         CLUSTER-IP      EXTERNAL-IP
 PORT(S)   AGE
service/eks-sample-linux-service   ClusterIP    10.100.74.8     <none>        80/
TCP    32m

NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/eks-sample-linux-deployment 3/3     3            3           27m

NAME                                                     DESIRED   CURRENT   READY
   AGE
replicaset.apps/eks-sample-linux-deployment-776d8f8fd8   3         3         3
   27m
```

Nell'output, è possibile visualizzare il servizio e l'implementazione specificati nel manifesto di esempio implementato nei passaggi precedenti. Saranno visualizzati anche tre Pods. Questo perché 3 `replicas` sono state specificate nel manifesto di esempio. Per ulteriori informazioni su Pods, consulta Pod nella documentazione di Kubernetes. Kubernetes crea automaticamente la risorsa `replicaset`, anche se non è specificato nei manifesti di esempio. Per ulteriori informazioni ReplicaSetin meritoReplicaSets, consulta la Kubernetes documentazione.

> **Note**
>
> Kubernetes mantiene il numero di repliche specificate nel manifesto. Se si tratta di un'implementazione di produzione e si desidera che Kubernetes dimensioni orizzontalmente il numero di repliche e dimensioni verticalmente le risorse di calcolo per i Pods, utilizza Horizontal Pod Autoscaler e Vertical Pod Autoscaler.

5. Visualizzare i dettagli del servizio implementato. Se hai implementato un servizio Windows, sostituisci *`linux`* con **`windows`**.

```
kubectl -n eks-sample-app describe service eks-sample-linux-service
```

Di seguito viene riportato un output di esempio:

Se hai implementato risorse Windows, allora tutte le istanze di *`linux`* nel seguente output saranno `windows`. Gli altri *`valori di esempio`* possono essere diversi dall'output.

```
Name:              eks-sample-linux-service
```

```
Namespace:          eks-sample-app
Labels:             app=eks-sample-linux-app
Annotations:        <none>
Selector:           app=eks-sample-linux-app
Type:               ClusterIP
IP Families:        <none>
IP:                 10.100.74.8
IPs:                10.100.74.8
Port:               <unset>  80/TCP
TargetPort:         80/TCP
Endpoints:          192.168.24.212:80,192.168.50.185:80,192.168.63.93:80
Session Affinity:   None
Events:             <none>
```

Nell'output precedente, il valore per `IP:` è un indirizzo IP univoco che può essere raggiunto da qualsiasi nodo o Pod all'interno del cluster ma non dall'esterno del cluster. I valori per `Endpoints` sono indirizzi IP assegnati all'interno del VPC ai Pods che fanno parte del servizio.

6. Visualizzare i dettagli di uno dei Pods elencati nell'output al momento della visualizzazione dello spazio dei nomi in una fase precedente. Se hai implementato un'app Windows, sostituisci *linux* con **windows** e *776d8f8fd8-78w66* con il valore restituito per uno dei tuoi Pods.

```
kubectl -n eks-sample-app describe pod eks-sample-linux-deployment-65b7669776-m6qxz
```

Output abbreviato

Se hai implementato risorse Windows, allora tutte le istanze di *linux* nel seguente output saranno `windows`. Gli altri *example values* possono essere diversi dall'output.

```
Name:         eks-sample-linux-deployment-65b7669776-m6qxz
Namespace:    eks-sample-app
Priority:     0
Node:         ip-192-168-45-132.us-west-2.compute.internal/192.168.45.132
[...]
IP:           192.168.63.93
IPs:
  IP:           192.168.63.93
Controlled By:  ReplicaSet/eks-sample-linux-deployment-65b7669776
[...]
Conditions:
  Type              Status
  Initialized       True
```

```
  Ready             True
  ContainersReady   True
  PodScheduled      True
[...]
Events:
  Type    Reason     Age    From
 Message
  ----    ------     ----   ----
 -------
  Normal  Scheduled  3m20s  default-scheduler
 Successfully assigned eks-sample-app/eks-sample-linux-deployment-65b7669776-m6qxz
 to ip-192-168-45-132.us-west-2.compute.internal
[...]
```

Nell'output precedente, il valore per `IP:` è un IP univoco assegnato al Pod dal blocco CIDR assegnato alla sottorete in cui si trova il nodo. Se preferisci che ai Pods vengano assegnati indirizzi IP da blocchi CIDR diversi, puoi modificare il comportamento di default. Per ulteriori informazioni, consulta Rete personalizzata per i pod. Puoi vedere anche come il pianificatore Kubernetesabbia pianificato il Pod sul `Node` con l'indirizzo IP *192.168.45.132*.

> **Tip**
>
> Anziché utilizzare la riga di comando, puoi visualizzare molti dettagli su Pods, servizi, implementazioni e altre risorse Kubernetes nella AWS Management Console. Per ulteriori informazioni, consulta Visualizzazione delle risorse Kubernetes.

7. Esegui una shell sul Pod descritto nel passaggio precedente, sostituendo *65b7669776-m6qxz* con l'ID di uno dei Pods.

   Linux

   ```
   kubectl exec -it eks-sample-linux-deployment-65b7669776-m6qxz -n eks-sample-app
    -- /bin/bash
   ```

   Windows

   ```
   kubectl exec -it eks-sample-windows-deployment-65b7669776-m6qxz -n eks-sample-
   app -- powershell.exe
   ```

8. Dalla shell del Pod, visualizza l'output dal server Web installato con l'implementazione in un passaggio precedente. È necessario specificare solo il nome del servizio. Questo viene risolto sull'indirizzo IP del servizio da CoreDNS, implementato con un cluster Amazon EKS, per impostazione predefinita.

   Linux

   ```
   curl eks-sample-linux-service
   ```

   Di seguito viene riportato un output di esempio:

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   [...]
   ```

   Windows

   ```
   Invoke-WebRequest -uri eks-sample-windows-service/default.html -UseBasicParsing
   ```

   Di seguito viene riportato un output di esempio:

   ```
   StatusCode        : 200
   StatusDescription : OK
   Content           : < h t m l > < b o d y > < b r / > < b r / > < m a r q u e e
    > < H 1 > H e l l o
                       E K S ! ! ! < H 1 > < m a r q u e e > < / b o d y > < h t
    m l >
   ```

9. Dalla shell del Pod, visualizza il server DNS per il Pod.

   Linux

   ```
   cat /etc/resolv.conf
   ```

   Di seguito viene riportato un output di esempio:

   ```
   nameserver 10.100.0.10
   ```

```
search eks-sample-app.svc.cluster.local svc.cluster.local cluster.local us-
west-2.compute.internal
options ndots:5
```

Nell'output precedente, `10.100.0.10` viene assegnato automaticamente come `nameserver` per tutti i Podsimplementati nel cluster.

Windows

```
Get-NetIPConfiguration
```

Output abbreviato

```
InterfaceAlias       : vEthernet
[...]
IPv4Address          : 192.168.63.14
[...]
DNSServer            : 10.100.0.10
```

Nell'output precedente, `10.100.0.10` viene assegnato automaticamente come il server DNS per tutti i Pods implementati nel cluster.

10. Disconnettersi dal Pod digitando `exit`.
11. Una volta terminato l'uso dell'applicazione di esempio, sarà possibile rimuovere lo spazio dei nomi, il servizio e l'implementazione di esempio con il seguente comando.

```
kubectl delete namespace eks-sample-app
```

Fasi successive

Dopo aver distribuito l'applicazione di esempio, potresti provare alcuni dei seguenti esercizi:

- the section called “Bilanciamento del carico di applicazione”
- the section called “Bilanciamento del carico di rete”

Vertical Pod Autoscaler

Il Vertical Pod Autoscaler di Kubernetes regola automaticamente le prenotazioni di CPU e memoria per i tuoi Pods in modo da "dimensionare correttamente" le tue applicazioni. Questo processo di adeguamento può migliorare l'utilizzo delle risorse del cluster e liberare CPU e memoria per altri Pods. Questo argomento illustra come implementare Vertical Pod Autoscaler nel cluster e verificarne il funzionamento.

Prerequisiti

- Il cluster Amazon EKS è già esistente. Se non lo è, consulta Guida introduttiva ad Amazon EKS.
- Kubernetes Metrics Server è già installato. Per ulteriori informazioni, consulta Installazione di Kubernetes Metrics Server.
- Stai utilizzando un client `kubectl` che è configurato per comunicare con il cluster Amazon EKS.
- OpenSSL `1.1.1` o versioni successive deve essere installato sul dispositivo.

Implementazione di Vertical Pod Autoscaler

In questa sezione, verrà implementato Vertical Pod Autoscaler nel cluster.

Implementazione di Vertical Pod Autoscaler

1. Aprire una finestra del terminale e passare a una directory in cui si desidera scaricare il codice sorgente di Vertical Pod Autoscaler.
2. Clona il repository GitHub kubernetes/autoscaler.

   ```
   git clone https://github.com/kubernetes/autoscaler.git
   ```

3. Passare alla directory `vertical-pod-autoscaler`.

   ```
   cd autoscaler/vertical-pod-autoscaler/
   ```

4. (Facoltativo) Se è già stata implementata un'altra versione di Vertical Pod Autoscaler, rimuoverla con il comando seguente.

   ```
   ./hack/vpa-down.sh
   ```

5. Se i nodi non hanno accesso Internet al registro del container `registry.k8s.io`, devi estrarre le seguenti immagini e inviarle al tuo repository privato. Per ulteriori informazioni su come estrarre e inviare immagini al tuo repository privato, consulta la sezione Copia di un'immagine di container da un repository a un altro.

```
registry.k8s.io/autoscaling/vpa-admission-controller:0.10.0
registry.k8s.io/autoscaling/vpa-recommender:0.10.0
registry.k8s.io/autoscaling/vpa-updater:0.10.0
```

   Se invii le immagini a un repository Amazon ECR privato, sostituisci `registry.k8s.io` nei manifesti con il tuo registro. Sostituisci *111122223333* con l'ID del tuo account. Sostituisci *region-code* con la Regione AWS in cui si trova il cluster. I comandi seguenti presuppongono che tu abbia dato al repository lo stesso nome del repository nel manifesto. Se hai dato al repository un nome diverso, dovrai modificare anch'esso.

```
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/admission-controller-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/recommender-deployment.yaml
sed -i.bak -e 's/registry.k8s.io/111122223333.dkr.ecr.region-
code.amazonaws.com/' ./deploy/updater-deployment.yaml
```

6. Implementare Vertical Pod Autoscaler nel cluster mediante il comando seguente.

```
./hack/vpa-up.sh
```

7. Verifica che i Pods di Vertical Pod Autoscaler siano stati creati correttamente.

```
kubectl get pods -n kube-system
```

   Di seguito viene riportato un output di esempio:

```
NAME                                        READY   STATUS    RESTARTS   AGE
[...]
metrics-server-8459fc497-kfj8w              1/1     Running   0          83m
vpa-admission-controller-68c748777d-ppspd   1/1     Running   0          7s
vpa-recommender-6fc8c67d85-gljpl            1/1     Running   0          8s
vpa-updater-786b96955c-bgp9d                1/1     Running   0          8s
```

Test dell'installazione di Vertical Pod Autoscaler

In questa sezione, verrà implementata un'applicazione di esempio per verificare il corretto funzionamento di Vertical Pod Autoscaler.

Per testare l'installazione di Vertical Pod Autoscaler

1. Implementare l'applicazione di esempio `hamster.yaml` per Vertical Pod Autoscaler mediante il comando seguente.

   ```
   kubectl apply -f examples/hamster.yaml
   ```

2. Recupera i Pods dall'applicazione di esempio `hamster`.

   ```
   kubectl get pods -l app=hamster
   ```

   Di seguito viene riportato un output di esempio:

   ```
   hamster-c7d89d6db-rglf5   1/1     Running   0          48s
   hamster-c7d89d6db-znvz5   1/1     Running   0          48s
   ```

3. Descrivi uno dei Pods per visualizzarne la relativa prenotazione di `cpu` e `memory`. Sostituisci *c7d89d6db-rglf5* con uno degli ID restituiti nell'output del passaggio precedente.

   ```
   kubectl describe pod hamster-c7d89d6db-rglf5
   ```

   Di seguito viene riportato un output di esempio:

   ```
   [...]
   Containers:
     hamster:
       Container ID:  docker://
   e76c2413fc720ac395c33b64588c82094fc8e5d590e373d5f818f3978f577e24
       Image:         registry.k8s.io/ubuntu-slim:0.1
       Image ID:      docker-pullable://registry.k8s.io/ubuntu-
   slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
       Port:          <none>
       Host Port:     <none>
       Command:
         /bin/sh
       Args:
   ```

```
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:35:16 -0700
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        100m
      memory:     50Mi
[...]
```

Tieni presente che il Pod originale riserva 100 millicpu di CPU e 50 mebibyte di memoria. Per questa applicazione di esempio, 100 millicpu è un valore inferiore a quanto il Pod deve eseguire, quindi risulta limitato dalla CPU. Il pod riserva inoltre una quantità di memoria nettamente inferiore rispetto alla necessaria. L'implementazione di `vpa-recommender` di Vertical Pod Autoscaler analizza i Pods `hamster` per verificare se i requisiti di CPU e memoria sono appropriati. Se sono necessari adeguamenti, `vpa-updater` riavvia i Podscon i valori aggiornati.

4. Attendi che `vpa-updater` avvii un nuovo Pod `hamster`. Potrebbero essere necessari uno o due minuti. Puoi monitorare i Pods con il seguente comando.

> **Note**
>
> Se non sei certo che sia stato avviato un nuovo Pod, confronta i nomi dei Pod facendo riferimento all'elenco precedente. All'avvio del nuovo Pod, verrà visualizzato un nuovo nome per il Pod.

```
kubectl get --watch Pods -l app=hamster
```

5. Quando viene avviato un nuovo pod Pod `hamster`, aggiungi una descrizione e visualizza le prenotazioni di CPU e memoria aggiornate.

```
kubectl describe pod hamster-c7d89d6db-jxgfv
```

Di seguito viene riportato un output di esempio:

```
[...]
Containers:
  hamster:
```

```
    Container ID:
 docker://2c3e7b6fb7ce0d8c86444334df654af6fb3fc88aad4c5d710eac3b1e7c58f7db
    Image:         registry.k8s.io/ubuntu-slim:0.1
    Image ID:      docker-pullable://registry.k8s.io/ubuntu-
slim@sha256:b6f8c3885f5880a4f1a7cf717c07242eb4858fdd5a84b5ffe35b1cf680ea17b1
    Port:          <none>
    Host Port:     <none>
    Command:
      /bin/sh
    Args:
      -c
      while true; do timeout 0.5s yes >/dev/null; sleep 0.5s; done
    State:          Running
      Started:      Fri, 27 Sep 2019 10:37:08 -0700
    Ready:          True
    Restart Count:  0
    Requests:
      cpu:        587m
      memory:     262144k
[...]
```

Nell'output precedente puoi osservare che la prenotazione della `cpu` è passata a 587 millicpu, ossia un valore di oltre cinque volte superiore a quello originale. La `memory` è aumentata a 262.144 KB (circa 250 mebibyte), ossia cinque volte il valore originale. Questo Pod disponeva di un numero insufficiente di risorse e pertanto Vertical Pod Autoscaler ha corretto la stima con un valore più adeguato.

6. Descrivere la risorsa `hamster-vpa` per visualizzare la nuova raccomandazione.

```
kubectl describe vpa/hamster-vpa
```

Di seguito viene riportato un output di esempio:

```
Name:         hamster-vpa
Namespace:    default
Labels:       <none>
Annotations:  kubectl.kubernetes.io/last-applied-configuration:
                {"apiVersion":"autoscaling.k8s.io/
v1beta2","kind":"VerticalPodAutoscaler","metadata":{"annotations":
{},"name":"hamster-vpa","namespace":"d...
API Version:  autoscaling.k8s.io/v1beta2
Kind:         VerticalPodAutoscaler
```

```
Metadata:
  Creation Timestamp:  2019-09-27T18:22:51Z
  Generation:          23
  Resource Version:    14411
  Self Link:           /apis/autoscaling.k8s.io/v1beta2/namespaces/default/
verticalpodautoscalers/hamster-vpa
  UID:                 d0d85fb9-e153-11e9-ae53-0205785d75b0
Spec:
  Target Ref:
    API Version:  apps/v1
    Kind:         Deployment
    Name:         hamster
Status:
  Conditions:
    Last Transition Time:  2019-09-27T18:23:28Z
    Status:                True
    Type:                  RecommendationProvided
  Recommendation:
    Container Recommendations:
      Container Name:  hamster
      Lower Bound:
        Cpu:     550m
        Memory:  262144k
      Target:
        Cpu:     587m
        Memory:  262144k
      Uncapped Target:
        Cpu:     587m
        Memory:  262144k
      Upper Bound:
        Cpu:     21147m
        Memory:  387863636
Events:          <none>
```

7. Dopo aver provato l'applicazione di esempio, è possibile eliminarla con il comando seguente.

```
kubectl delete -f examples/hamster.yaml
```

Horizontal Pod Autoscaler

Horizontal Pod Autoscaler di Kubernetes dimensiona automaticamente il numero di Pods in un'implementazione, un controller di replica o un set di repliche in base all'utilizzo della CPU di tale

risorsa. Ciò consente alle applicazioni di dimensionare orizzontalmente le risorse in base all'aumento della domanda, o di ridurle orizzontalmente quando non sono necessarie, liberando così i nodi per altre applicazioni. Quando imposti una percentuale di utilizzo della CPU target, Horizontal Pod Autoscaler aumenta o riduce le risorse dell'applicazione in modo conforme all'obiettivo prefissato.

Horizontal Pod Autoscaler è una risorsa API standard in Kubernetes il cui funzionamento richiede solo l'installazione di una fonte di parametri (ad esempio il server di parametri Kubernetes) nel cluster Amazon EKS. Non è necessario implementare o installare Horizontal Pod Autoscaler sul cluster per iniziare a dimensionare le applicazioni. Per ulteriori informazioni, consulta Horizontal Pod Autoscaler nella documentazione Kubernetes.

Utilizza questo argomento per preparare Horizontal Pod Autoscaler per il cluster Amazon EKS e per verificare che funzioni con un'applicazione di esempio.

> **Note**
>
> Questo argomento si basa sulla procedura guidata per Horizontal Pod autoscaler riportata nella documentazione di Kubernetes.

Prerequisiti

- Il cluster Amazon EKS è già esistente. Se non lo è, consulta Guida introduttiva ad Amazon EKS.
- Kubernetes Metrics Server è già installato. Per ulteriori informazioni, consulta Installazione di Kubernetes Metrics Server.
- Stai utilizzando un client `kubectl` che è configurato per comunicare con il cluster Amazon EKS.

Esecuzione di un'applicazione di test di Horizontal Pod Autoscaler

In questa sezione implementerai un'applicazione di esempio per verificare il corretto funzionamento di Horizontal Pod Autoscaler.

> **Note**
>
> Questo esempio si basa sulla procedura guidata per Horizontal Pod Autoscaler riportata nella documentazione Kubernetes.

Per testare l'installazione di Horizontal Pod Autoscaler

1. Implementa un'applicazione server Web Apache con il comando seguente.

```
kubectl apply -f https://k8s.io/examples/application/php-apache.yaml
```

   Questo Pod del server Web Apache ha un limite di CPU di 500 millicpu e serve sulla porta 80.

2. Crea una risorsa Horizontal Pod Autoscaler per l'implementazione di `php-apache`.

```
kubectl autoscale deployment php-apache --cpu-percent=50 --min=1 --max=10
```

   Questo comando crea un componente di scalabilità automatica che destina il 50% di utilizzo della CPU per l'implementazione, con un minimo di un Pod e un massimo di dieci Pods. Quando il carico medio della CPU è inferiore al 50%, tale componente cerca di ridurre il numero di Pods nell'implementazione, fino a un minimo di uno. Quando il carico è maggiore del 50%, il componente di scalabilità automatica cerca di aumentare il numero di Pods nell'implementazione, fino a un massimo di dieci. Per ulteriori informazioni, consulta How does HorizontalPodAutoscaler a? nella Kubernetes documentazione.

3. Descrivere il componente di scalabilità automatica con il comando seguente per visualizzarne i dettagli.

```
kubectl get hpa
```

   Di seguito viene riportato un output di esempio:

```
NAME         REFERENCE                TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   0%/50%    1         10        1          51s
```

   Come si può vedere, il carico corrente della CPU è `0%` perché non c'è ancora carico sul server. Il conteggio dei Pod è già al limite più basso (uno), quindi non può essere ridotto orizzontalmente.

4. Crea un carico per il server Web eseguendo un container.

```
kubectl run -i \
    --tty load-generator \
    --rm --image=busybox \
    --restart=Never \
    -- /bin/sh -c "while sleep 0.01; do wget -q -O- http://php-apache; done"
```

5. Per osservare l'incremento dell'implementazione, esegui periodicamente il seguente comando in un terminale separato dal terminale in cui hai eseguito il passaggio precedente.

```
kubectl get hpa php-apache
```

Di seguito viene riportato un output di esempio:

```
NAME         REFERENCE                  TARGETS    MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   250%/50%   1         10        5
 4m44s
```

Potrebbe essere necessario più di un minuto per aumentare il numero di repliche. Finché la percentuale effettiva della CPU è superiore alla percentuale di destinazione, il conteggio delle repliche aumenta fino a 10. In questo caso, è `250%`, quindi il numero di `REPLICAS` continua ad aumentare.

> **Note**
>
> Potrebbero essere necessari alcuni minuti prima che il numero di repliche raggiunga il limite massimo. Se, ad esempio, sono necessarie solo 6 repliche affinché il carico della CPU rimanga pari o inferiore al 50%, il carico non verrà dimensionato oltre 6 repliche.

6. Arresta il carico. Nella finestra del terminale in cui si sta generando il carico, arrestare il carico tenendo premuti i tasti `Ctrl+C`. È possibile guardare il dimensionamento delle repliche nuovamente a 1 eseguendo di nuovo il seguente comando nel terminale corrispondente.

```
kubectl get hpa
```

Di seguito viene riportato un output di esempio:

```
NAME         REFERENCE                  TARGETS   MINPODS   MAXPODS   REPLICAS   AGE
php-apache   Deployment/php-apache   0%/50%    1         10        1          25m
```

> **Note**
>
> Il periodo di tempo di default per il ridimensionamento è di cinque minuti, quindi ci vorrà un po' di tempo prima di vedere il conteggio delle repliche raggiungere di nuovo 1, anche quando la percentuale corrente della CPU è 0. L'intervallo di tempo è modificabile.

Per ulteriori informazioni, consulta Horizontal Pod Autoscaler nella documentazione Kubernetes.

7. Al termine dell'esercitazione con l'applicazione di esempio, elimina le risorse `php-apache`.

```
kubectl delete deployment.apps/php-apache service/php-apache
 horizontalpodautoscaler.autoscaling/php-apache
```

Bilanciamento del carico di rete su Amazon EKS

Il traffico di rete è bilanciato al carico `L4` del modello OSI. Per bilanciare il traffico delle applicazioni in`L7`, si distribuisce un Kubernetes`ingress`, che fornisce un AWS Application Load Balancer. Per ulteriori informazioni, consulta Bilanciamento del carico di applicazione su Amazon EKS. Per ulteriori informazioni sulle differenze tra i due tipi di bilanciamento del carico, consulta le funzionalità di Elastic Load Balancing sul AWS sito Web.

Quando crei un sistema di Kubernetes`Service` bilanciamento del carico`LoadBalancer`, il controller di bilanciamento del carico del provider di servizi AWS cloud crea AWSClassic Load Balancer per impostazione predefinita, ma può anche creare Network Load Balancer. AWS Questo controller in futuro riceverà solo correzioni di bug critici. Per ulteriori informazioni sull'utilizzo del load balancer del provider AWS cloud, consulta il controller di bilanciamento del carico del provider AWS cloud nella documentazione. Kubernetes Il suo utilizzo non è descritto in questo argomento.

Consigliamo di utilizzare la versione `2.7.2` o una versione successiva di AWS Load Balancer Controller anziché il controller del load balancer del provider del cloud AWS . AWS Load Balancer ControllerCrea AWS Network Load Balancer, ma non crea AWS Classic Load Balancer. Il resto di questo argomento riguarda l'utilizzo del AWS Load Balancer Controller.

Un AWS Network Load Balancer può bilanciare il carico del traffico di rete da Pods distribuire su destinazioni IP e istanze di Amazon EC2 o su destinazioni IP. AWS Fargate Per ulteriori informazioni, consulta Controller del load balancer AWS su GitHub.

Prerequisiti

Prima di eseguire il bilanciamento del carico del traffico di rete utilizzando il AWS Load Balancer Controller, è necessario soddisfare i seguenti requisiti.

- Avere un cluster esistente. Se non si dispone di un cluster esistente, consultare Guida introduttiva ad Amazon EKS. Se è necessario aggiornare la versione di un cluster esistente, vedere Aggiornamento della versione di Kubernetes del cluster Amazon EKS.
- Aver implementato AWS Load Balancer Controller nel cluster. Per ulteriori informazioni, consulta Che cosa è la AWS Load Balancer Controller?. Consigliamo la versione `2.7.2` o successiva.
- Disporre di almeno una sottorete. Se si trovano più sottoreti con tag in una zona di disponibilità, il controller sceglie la sottorete il cui ID sottorete viene prima in ordine lessicografico. La sottorete deve avere a disposizione almeno otto indirizzi IP.
- Se si utilizza il AWS Load Balancer Controller versione `2.1.1` o precedente, le sottoreti devono essere taggate come segue. Se si utilizza la versione `2.1.2` o successiva, questo tag è facoltativo. Potresti voler etichettare una sottorete se hai più cluster in esecuzione nello stesso VPC o più AWS servizi che condividono sottoreti in un VPC e desideri un maggiore controllo su dove vengono forniti i load balancer per ogni cluster. Se specifichi in modo esplicito gli ID sottorete come annotazione su un oggetto di servizio, Kubernetes e il AWS Load Balancer Controller utilizzano queste sottoreti direttamente per creare il load balancer. Il tagging delle sottoreti non è obbligatorio se si sceglie di utilizzare questo metodo per il provisioning dei load balancer ed è possibile ignorare i seguenti requisiti di assegnazione di tag delle sottoreti pubbliche e private. Sostituisci *my-cluster* con il nome del cluster.
  - Chiave: `kubernetes.io/cluster/`*my-cluster*
  - Valore: `shared` o `owned`
- Le sottoreti pubbliche e private devono soddisfare i seguenti requisiti, a meno che non si specifichino esplicitamente gli ID sottorete come annotazione in un oggetto di servizio o di ingresso. Se effettui il provisioning dei load balancer specificando esplicitamente gli ID sottorete come annotazione in un oggetto di servizio o di ingresso, Kubernetes e il AWS Load Balancer Controller utilizzano queste sottoreti direttamente per creare il load balancer e non sono quindi necessari i seguenti tag.
  - Sottoreti private: deve essere taggato nel seguente formato. In questo modo il AWS Load Balancer Controller sa che le sottoreti possono essere utilizzate per bilanciatori di carico interni. Kubernetes Se utilizzi `eksctl` o un AWS AWS CloudFormation modello Amazon EKS per creare il tuo VPC dopo il 26 marzo 2020, le sottoreti vengono etichettate in modo appropriato al momento della creazione. Per ulteriori informazioni sui modelli AWS AWS CloudFormation VPC Amazon EKS, vedere Creazione di un VPC per il cluster Amazon EKS.
    - Chiave: `kubernetes.io/role/internal-elb`
    - Valore: 1

- Sottoreti pubbliche: deve essere taggato nel seguente formato. In questo modo, Kubernetes utilizza solo tali sottoreti per load balancer esterni, anziché scegliere una sottorete pubblica in ciascuna zona di disponibilità (in ordine lessicografico per ID sottorete). Se utilizzi `eksctl` o un AWS CloudFormation modello Amazon EKS per creare il tuo VPC dopo il 26 marzo 2020, le sottoreti vengono etichettate in modo appropriato al momento della creazione. Per ulteriori informazioni sui modelli AWS CloudFormation VPC di Amazon EKS, consulta. Creazione di un VPC per il cluster Amazon EKS
  - Chiave: `kubernetes.io/role/elb`
  - Valore: 1

Se i tag del ruolo della sottorete non vengono aggiunti in modo esplicito, il controller del servizio Kubernetes esamina la tabella di instradamento delle sottoreti VPC del cluster per determinare se la sottorete è privata o pubblica. Si consiglia di non fare affidamento su questo comportamento e di aggiungere esplicitamente i tag di ruolo pubblici o privati. Il AWS Load Balancer Controller non esamina le tabelle di instradamento e richiede che i tag privati e pubblici siano presenti per il rilevamento automatico corretto.

Considerazioni

- La configurazione del load balancer è controllata da annotazioni che vengono aggiunte al manifest del servizio. Le annotazioni del servizio sono diverse quando si utilizza il AWS Load Balancer Controller rispetto a quando si utilizza il controller di bilanciamento del carico del provider di AWS cloud. Assicurarsi di rivedere le annotazioni per il AWS Load Balancer Controller prima di implementare i servizi.
- Quando utilizzi Amazon VPC CNI plugin for Kubernetes, il AWS Load Balancer Controller è in grado di bilanciare il carico su destinazioni IP o di istanza Amazon EC2 e su destinazioni IP Fargate. Quando utilizzi plug-in CNI compatibili alternativi, il controller può bilanciare il carico solo per le destinazioni istanza. Per ulteriori informazioni sui tipi di destinazioni dei Network Load Balancer, consultare Target type (Tipo di destinazione) nella Guida per l'utente di Network Load Balancer.
- Se si desidera aggiungere tag al load balancer quando o dopo la sua creazione, aggiungere la seguente annotazione nella specifica del servizio. Per ulteriori informazioni, consulta Tag delle risorse AWS nella documentazione di AWS Load Balancer Controller.

```
service.beta.kubernetes.io/aws-load-balancer-additional-resource-tags
```

- È possibile assegnare Indirizzi IP elastici al Network Load Balancer aggiungendo la seguente annotazione. Sostituire i *example values* con gli Allocation IDs degli indirizzi IP elastici. Il numero di Allocation IDs deve corrispondere al numero di sottoreti utilizzate per il load balancer. Per ulteriori informazioni, consulta la documentazione AWS Load Balancer Controller.

```
service.beta.kubernetes.io/aws-load-balancer-eip-allocations:
 eipalloc-xxxxxxxxxxxxxxxxx,eipalloc-yyyyyyyyyyyyyyyyy
```

- Amazon EKS aggiunge una regola in entrata al gruppo di sicurezza del nodo per il traffico client e una regola per ogni sottorete del sistema di bilanciamento del carico nel VPC per i controlli dell'integrità per ogni Network Load Balancer creato. L'implementazione di un servizio di tipo LoadBalancer può non riuscire se Amazon EKS tenta di creare regole che superano la quota per il numero massimo di regole consentite per un gruppo di sicurezza. Per ulteriori informazioni, consultare Gruppi di sicurezza in Quote di Amazon VPC nella Guida per l'utente di Amazon VPC. Considerare le opzioni seguenti per ridurre al minimo le possibilità di superare il numero massimo di regole per un gruppo di sicurezza:
  - Richiedere un aumento delle regole per ogni quota di gruppo di sicurezza. Per ulteriori informazioni, consultare Richiesta di un aumento di quota nella Guida per l'utente di Service Quotas.
  - Al posto di destinazioni di istanza, utilizzare destinazioni IP. Con le destinazioni IP, è possibile condividere le regole per le stesse porte di destinazione. È possibile specificare manualmente le sottoreti del load balancer con un'annotazione. Per ulteriori informazioni, consulta Annotazioni su GitHub.
  - Utilizzare un ingresso invece di un servizio di tipo LoadBalancer per inviare traffico al tuo servizio. L' AWS Application Load Balancer richiede meno regole rispetto ai Network Load Balancer. È possibile condividere un ALB su più ingressi. Per ulteriori informazioni, consulta Bilanciamento del carico di applicazione su Amazon EKS. Non è possibile condividere un Network Load Balancer su più servizi.
  - Implementa i cluster su più account.
- Se i Pods vengono eseguiti su Windows in un cluster Amazon EKS, un singolo servizio con un load balancer può supportare fino a 1024 Pods di back-end. Ogni Pod ha il proprio indirizzo IP univoco.
- Si consiglia di creare solo nuovi Network Load Balancer con il AWS Load Balancer Controller. Il tentativo di sostituire i Network Load Balancer esistenti creati con il controller di bilanciamento del carico del provider di servizi AWS cloud può comportare la creazione di più Network Load Balancer che potrebbero causare tempi di inattività delle applicazioni.

Creazione di un Network Load Balancer

È possibile creare un Network Load Balancer con destinazioni IP o istanza.

IP targets

Puoi utilizzare le destinazioni IP con Pods implementati su nodi Amazon EC2 o Fargate. Il servizio Kubernetes deve essere creato come tipo `LoadBalancer`. Per ulteriori informazioni, consulta Type nella documentazione. LoadBalancer Kubernetes

Per creare un load balancer che utilizzi destinazioni IP, aggiungere le seguenti annotazioni a un manifesto del servizio e implementare il servizio. Il `external` valore per `aws-load-balancer-type` è ciò che fa sì che il controller del load balancer del provider AWS cloudAWS Load Balancer Controller, anziché il controller del servizio cloud, crei il Network Load Balancer. É possibile visualizzare un manifesto del servizio di esempio con le annotazioni.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "ip"
```

> **Note**
>
> Se stai eseguendo il bilanciamento del carico sui Pods `IPv6`, aggiungi la seguente annotazione alla specifica. È possibile bilanciare il carico solo su destinazioni da `IPv6` a IP, non su destinazioni di istanza. Senza questa annotazione, il bilanciamento del carico è su `IPv4`.
>
> ```
> service.beta.kubernetes.io/aws-load-balancer-ip-address-type: dualstack
> ```

Per impostazione predefinita, i Network Load Balancer vengono creati con `internal` `aws-load-balancer-scheme`. Puoi avviare i Network Load Balancer in qualunque sottorete nel VPC del tuo cluster, incluse sottoreti non specificate quando hai creato il cluster.

Kubernetes esamina la tabella di instradamento delle sottoreti per stabilire se sono pubbliche o private. Le sottoreti pubbliche hanno una route direttamente a Internet tramite un gateway Internet; non vale altrettanto per le sottoreti private.

Se si desidera creare un Network Load Balancer in una sottorete pubblica per il bilanciamento del carico sui nodi Amazon EC2 (Fargate può essere solo privato), specificare `internet-facing` con la seguente annotazione:

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Note**
>
> Al fine di garantire la compatibilità con le versioni precedenti, l'annotazione `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"` è ancora supportata. Consigliamo, tuttavia, di utilizzare le annotazioni precedenti per i load balancer anziché `service.beta.kubernetes.io/aws-load-balancer-type: "nlb-ip"`.

> **Important**
>
> Non modificare le annotazioni dopo aver creato il servizio. Se è necessario modificarlo, eliminare l'oggetto di servizio e crearlo nuovamente con il valore desiderato per questa annotazione.

Instance targets

Il controller di bilanciamento del carico del provider AWS cloud crea Network Load Balancer solo con obiettivi di istanza. La versione `2.2.0` e successive di AWS Load Balancer Controller crea anche Network Load Balancer con destinazioni di istanza. Ti consigliamo di utilizzarlo, anziché il controller di bilanciamento del carico del provider AWS cloud, per creare nuovi Network Load Balancer. Puoi utilizzare le destinazioni di istanza di Network Load Balancer con Pods implementati sui nodi Amazon EC2, ma non in Fargate. Per bilanciare il carico del traffico di rete tra i Pods implementati in Fargate, è necessario utilizzare destinazioni IP.

Per implementare un Network Load Balancer in una sottorete privata, la specifica del servizio deve avere le seguenti annotazioni. É possibile visualizzare un manifesto del servizio di esempio con le annotazioni. Il `external` valore per `aws-load-balancer-type` è ciò che fa sì che il AWS Load Balancer Controller, anziché il controller del load balancer del provider AWS cloud, crei il Network Load Balancer.

```
service.beta.kubernetes.io/aws-load-balancer-type: "external"
```

```
service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: "instance"
```

Per impostazione predefinita, i Network Load Balancer vengono creati con `internal` `aws-load-balancer-scheme`. Per i Network Load Balancer interni, il cluster Amazon EKS deve essere configurato per l'utilizzo di almeno una sottorete privata nel VPC. Kubernetes esamina latabella di instradamento delle sottoreti per stabilire se sono pubbliche o private. Le sottoreti pubbliche hanno una route direttamente a Internet tramite un gateway Internet; non vale altrettanto per le sottoreti private.

Se si desidera creare un Network Load Balancer in una sottorete pubblica per il bilanciamento del carico sui nodi Amazon EC2, specificare `internet-facing` con la seguente annotazione:

```
service.beta.kubernetes.io/aws-load-balancer-scheme: "internet-facing"
```

> **Important**
>
> Non modificare le annotazioni dopo aver creato il servizio. Se è necessario modificarlo, eliminare l'oggetto di servizio e crearlo nuovamente con il valore desiderato per questa annotazione.

(Facoltativo) Implementare un'applicazione di esempio

Prerequisiti

- Almeno una sottorete pubblica o privata nel VPC del cluster.
- Aver implementato AWS Load Balancer Controller nel cluster. Per ulteriori informazioni, consulta Che cosa è la AWS Load Balancer Controller?. Consigliamo la versione `2.7.2` o successiva.

Per implementare un'applicazione di esempio

1. Se stai eseguendo l'implementazione su Fargate, assicurati di avere una sottorete privata disponibile nel VPC e di creare un profilo Fargate. Se non stai eseguendo l'implementazione su Fargate, salta questa fase. É possibile creare il profilo eseguendo il seguente comando o nella AWS Management Console utilizzando gli stessi valori per `name` e `namespace` che si trovano nel comando. Sostituisci i *`example values`* con i valori in tuo possesso.

   ```
   eksctl create fargateprofile \
   ```

```
    --cluster my-cluster \
    --region region-code \
    --name nlb-sample-app \
    --namespace nlb-sample-app
```

2. Implementa un'applicazione di esempio.

   a. Crea uno spazio dei nomi per l'applicazione.

   ```
   kubectl create namespace nlb-sample-app
   ```

   b. Salva nel computer i seguenti contenuti in un file denominato *sample-deployment*.yaml.

   ```
   apiVersion: apps/v1
   kind: Deployment
   metadata:
     name: nlb-sample-app
     namespace: nlb-sample-app
   spec:
     replicas: 3
     selector:
       matchLabels:
         app: nginx
     template:
       metadata:
         labels:
           app: nginx
       spec:
         containers:
           - name: nginx
             image: public.ecr.aws/nginx/nginx:1.23
             ports:
               - name: tcp
                 containerPort: 80
   ```

   c. Applica il file manifesto al cluster.

   ```
   kubectl apply -f sample-deployment.yaml
   ```

3. Crea un servizio con un Network Load Balancer connesso a Internet che bilanci il carico sulle destinazioni IP.

a. Salva nel computer i seguenti contenuti in un file denominato *sample-service*`.yaml`. Se stai eseguendo l'implementazione sui nodi Fargate, rimuovi la riga `service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing`.

```
apiVersion: v1
kind: Service
metadata:
  name: nlb-sample-service
  namespace: nlb-sample-app
  annotations:
    service.beta.kubernetes.io/aws-load-balancer-type: external
    service.beta.kubernetes.io/aws-load-balancer-nlb-target-type: ip
    service.beta.kubernetes.io/aws-load-balancer-scheme: internet-facing
spec:
  ports:
    - port: 80
      targetPort: 80
      protocol: TCP
  type: LoadBalancer
  selector:
    app: nginx
```

b. Applica il file manifesto al cluster.

```
kubectl apply -f sample-service.yaml
```

4. Verificare che il servizio sia stato implementato.

```
kubectl get svc nlb-sample-service -n nlb-sample-app
```

Di seguito viene riportato un output di esempio:

```
NAME             TYPE           CLUSTER-IP         EXTERNAL-IP
                                                  PORT(S)        AGE
sample-service   LoadBalancer   10.100.240.137
 k8s-nlbsampl-nlbsampl-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
  80:32400/TCP   16h
```

> **Note**
>
> Regione AWS

5. Apri Amazon EC2 AWS Management Console. Seleziona Target Groups (Gruppi di destinazioni) in Load Balancing (Bilanciamento del carico) nel pannello di navigazione a sinistra. Nella colonna Nome, seleziona il nome del gruppo di destinazione in cui il valore nella colonna load balancer corrisponde al nome nella colonna `EXTERNAL-IP` dell'output nella fase precedente. Ad esempio, se l'output era uguale all'output indicato sopra sarà necessario selezionare il gruppo di destinazione denominato `k8s-default-samplese-`*`xxxxxxxxxx`*. Target type (Tipo di destinazione) è `IP` perché è stato specificato nel manifesto del servizio di esempio.
6. Selezionare il proprio Gruppo di destinazione e poi scegliere la scheda Destinazioni. In Destinazioni registrate, verranno visualizzati tre indirizzi IP delle tre repliche implementate in un passaggio precedente. Prima di continuare, attendere fino a quando lo stato di tutti gli obiettivi è Integro. Potrebbero passare diversi minuti prima che tutti gli obiettivi siano `healthy`. Gli obiettivi potrebbero essere in uno stato `unhealthy` prima di passare a uno stato `healthy`.
7. Invia il traffico al servizio sostituendo *`xxxxxxxxxx-xxxxxxxxxxxxxxxx`* e *`us-west-2`* con i valori restituiti nell'output di un operazione precedente per `EXTERNAL-IP`. Se è stata eseguita l'implementazione a una sottorete privata, devi visualizzare la pagina da un dispositivo all'interno del VPC, ad esempio un bastion host. Per ulteriori informazioni, consulta Host bastione Linux su AWS.

   ```
   curl k8s-default-samplese-xxxxxxxxxx-xxxxxxxxxxxxxxxx.elb.region-code.amazonaws.com
   ```

   Di seguito viene riportato un output di esempio:

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   [...]
   ```
8. Al termine dell'implementazione, del servizio e dello spazio dei nomi di esempio, rimuoverli.

   ```
   kubectl delete namespace nlb-sample-app
   ```

Bilanciamento del carico di applicazione su Amazon EKS

Quando si crea un Kubernetes`ingress`, viene eseguito il provisioning di un AWS Application Load Balancer (ALB) che bilancia il carico del traffico dell'applicazione. Per ulteriori informazioni, consulta Cos'è un Application Load Balancer? nella Guida per l'utente di Application Load Balancer e Ingressi nella documentazione di Kubernetes. Gli ALB possono essere utilizzati con Pods implementati sui nodi o su AWS Fargate. È possibile implementare un ALB a sottoreti pubbliche o private.

Il traffico delle applicazioni è bilanciato a `L7` del modello OSI. Per bilanciare il carico del traffico di rete su `L4`, implementa un `service` Kubernetes di tipo `LoadBalancer`. Questo tipo fornisce un AWS Network Load Balancer. Per ulteriori informazioni, consulta Bilanciamento del carico di rete su Amazon EKS. Per ulteriori informazioni sulle differenze tra i due tipi di bilanciamento del carico, consulta le caratteristiche di Elastic Load Balancing sul sito Web AWS .

Prerequisiti

Prima di poter bilanciare il carico del traffico dell'applicazione in un'applicazione, è necessario soddisfare i seguenti requisiti.

- Avere un cluster esistente. Se non si dispone di un cluster esistente, consultare Guida introduttiva ad Amazon EKS. Se è necessario aggiornare la versione di un cluster esistente, vedere Aggiornamento della versione di Kubernetes del cluster Amazon EKS.
- Aver implementato AWS Load Balancer Controller nel cluster. Per ulteriori informazioni, consulta Che cosa è la AWS Load Balancer Controller?. Consigliamo la versione `2.7.2` o successiva.
- Almeno due sottoreti devono trovarsi in diverse zone di disponibilità. Il AWS Load Balancer Controller sceglie una sottorete per ogni zona di disponibilità. Quando si trovano più sottoreti taggate in una zona di disponibilità, il controller sceglie la sottorete il cui ID sottorete viene prima in ordine lessicografico. Ogni sottorete deve avere a disposizione almeno otto indirizzi IP.

  Se si utilizzano più gruppi di sicurezza collegati al nodo (worker), esattamente un gruppo di sicurezza deve essere taggato come segue. Sostituisci *`my-cluster`* con il nome del cluster.

  - Chiave: `kubernetes.io/cluster/`*`my-cluster`*
  - Valore: `shared` o `owned`
- Se si utilizza il AWS Load Balancer Controller versione `2.1.1` o precedenti, le sottoreti devono essere taggate nel formato seguente. Se si utilizza la versione `2.1.2` o successiva, l'assegnazione di tag è facoltativa. Tuttavia, si consiglia di taggare una sottorete nel caso si verifichi una delle seguenti situazioni. Hai più cluster in esecuzione nello stesso VPC o AWS più servizi che

condividono sottoreti in un VPC. Oppure, si desidera un maggiore controllo sulla posizione in cui viene eseguito il provisioning dei bilanciatori di carico per ciascun cluster. Sostituisci *my-cluster* con il nome del cluster.

- Chiave: `kubernetes.io/cluster/`*my-cluster*
- Valore – `shared` o `owned`

- Le sottoreti pubbliche e private devono soddisfare i seguenti requisiti. Ciò avviene a condizione che non si indichino esplicitamente gli ID sottorete come annotazione in un oggetto di servizio o di ingresso. Si supponga di effettuare il provisioning dei load balancer specificando esplicitamente gli ID sottorete come annotazione in un oggetto di servizio o di ingresso. In questa situazione, Kubernetes e il controller del load balancer AWS utilizzeranno queste sottoreti direttamente per creare il load balancer senza che siano necessari i seguenti tag.
  - Sottoreti private: deve essere taggato nel seguente formato. In questo modo Kubernetes e il controller del AWS load balancer sanno che le sottoreti possono essere utilizzate per bilanciamenti del carico interni. Se utilizzi `eksctl` o un AWS CloudFormation modello Amazon EKS per creare il tuo VPC dopo il 26 marzo 2020, le sottoreti vengono etichettate in modo appropriato al momento della creazione. Per ulteriori informazioni sui modelli AWS CloudFormation VPC di Amazon EKS, consulta. Creazione di un VPC per il cluster Amazon EKS
    - Chiave: `kubernetes.io/role/internal-elb`
    - Valore: 1
  - Sottoreti pubbliche: deve essere taggato nel seguente formato. In questo modo Kubernetes sa di utilizzare solo le sottoreti specificate per i load balancer esterni. In questo modo, Kubernetes non sceglie una sottorete pubblica in ogni zona di disponibilità (in base all'ordine lessicografico degli ID sottorete). Se utilizzi `eksctl` o un AWS CloudFormation modello Amazon EKS per creare il tuo VPC dopo il 26 marzo 2020, le sottoreti vengono etichettate in modo appropriato al momento della creazione. Per ulteriori informazioni sui modelli AWS CloudFormation VPC di Amazon EKS, consulta. Creazione di un VPC per il cluster Amazon EKS
    - Chiave: `kubernetes.io/role/elb`
    - Valore: 1

  Se i tag del ruolo della sottorete non vengono aggiunti in modo esplicito, il controller del servizio Kubernetes esamina la tabella di instradamento delle sottoreti VPC del cluster. Ciò consente di determinare se la sottorete è privata o pubblica. Si consiglia di non fare affidamento su questo comportamento. É preferibile, invece, aggiungere esplicitamente i tag del ruolo privato o pubblico. Il AWS Load Balancer Controller non esamina le tabelle di instradamento. Richiede inoltre la presenza di tag privati e pubblici per un efficiente rilevamento automatico.

Considerazioni

- Il AWS Load Balancer Controller crea ALB e le AWS risorse di supporto necessarie ogni volta che viene creata una risorsa in Kubernetes ingresso sul cluster con l'annotazione. `kubernetes.io/ingress.class: alb` La risorsa di ingresso configura l'ALB per instradare il traffico HTTP o HTTPS verso Pods differenti all'interno del cluster. Per fare in modo che gli oggetti di ingresso utilizzino il AWS Load Balancer Controller, aggiungi la seguente annotazione alla specifica di ingresso Kubernetes. Per ulteriori informazioni, consulta Specifiche di ingresso su GitHub.

```
annotations:
    kubernetes.io/ingress.class: alb
```

> **Note**
>
> Se stai bilanciando il carico sui Pods `IPv6`, aggiungi la seguente annotazione alla specifica di ingresso. È possibile bilanciare il carico solo su destinazioni da `IPv6` a IP, non su destinazioni di istanza. Senza questa annotazione, il bilanciamento del carico è su `IPv4`.
>
> ```
> alb.ingress.kubernetes.io/ip-address-type: dualstack
> ```

- Il AWS Load Balancer Controller supporta le modalità di traffico seguenti:
  - Istanza: registra i nodi all'interno del cluster come destinazioni per l'ALB. Il traffico che raggiunge l'ALB viene indirizzato a `NodePort` per il tuo servizio e quindi inoltrato ai Pods. Questa è la modalità di traffico predefinita. È anche possibile specificarla esplicitamente con l'annotazione `alb.ingress.kubernetes.io/target-type: instance`.

    > **Note**
    >
    > Il Kubernetes servizio deve specificare il tipo `NodePort` o "LoadBalancer" per utilizzare questa modalità di traffico.

  - IP: registra Pods come destinazioni per l'ALB. Il traffico che raggiunge l'ALB viene indirizzato direttamente ai Pods per il servizio. È necessario specificare l'annotazione `alb.ingress.kubernetes.io/target-type: ip` per utilizzare questa modalità di traffico. Il tipo di destinazione IP è richiesta quando i Pods di destinazione sono in esecuzione su Fargate.

- Per assegnare i tag agli ALB creati dal controller, aggiungere la seguente annotazione al controller: `alb.ingress.kubernetes.io/tags`. Per un elenco di tutte le annotazioni disponibili supportate dal AWS Load Balancer Controller, consulta Annotazioni in ingresso su GitHub.
- L'aggiornamento o il downgrade della versione del controller ALB può introdurre modifiche importanti alle funzionalità che si basano su di essa. Per ulteriori informazioni sulle modifiche che vengono introdotte in ogni versione, consulta le note di rilascio del controller ALB su GitHub.

Condivisione di un Application Load Balancer su più risorse di servizio tramite **`IngressGroups`**

Per aggiungere un ingresso a un gruppo, aggiungi la seguente annotazione a una specifica di risorsa di ingresso Kubernetes.

```
alb.ingress.kubernetes.io/group.name: my-group
```

Il nome del gruppo deve:

- Avere una lunghezza pari o inferiore a 63 caratteri.
- Essere formato da lettere minuscole, numeri, `-`, e `.`
- Iniziare e finire con una lettera o un numero.

Il controller unisce automaticamente le regole di ingresso per tutti gli ingressi nello stesso gruppo di ingresso. Le supporta con un singolo ALB. La maggior parte delle annotazioni definite in un ingresso si applica solo ai percorsi definiti da tale ingresso. Per impostazione predefinita, le risorse in ingresso non appartengono ad alcun gruppo di ingresso.

> **Warning**
>
> Possibili rischi per la sicurezza: specifica un gruppo di ingressi per un ingresso solo quando tutti gli utenti Kubernetes che dispongono dell'autorizzazione RBAC per creare o modificare le risorse di ingresso rientrano nello stesso limite di attendibilità. Se si aggiunge l'annotazione con un nome di gruppo, altri utenti Kubernetes potrebbero creare o modificare i propri ingressi così da appartenere allo stesso gruppo di ingressi. Ciò può causare comportamenti indesiderati, ad esempio la sovrascrittura delle regole esistenti con regole con priorità più alta.

É possibile aggiungere un numero d'ordine della tua risorsa di ingresso.

```
alb.ingress.kubernetes.io/group.order: '10'
```

Il numero può variare tra 1 e 1000. Viene considerato prima il numero più basso per tutti gli ingressi nello stesso gruppo di ingressi. Tutti gli ingressi senza questa annotazione vengono valutati con un valore pari a zero. La duplicazione delle regole con un numero più alto può causare la sovrascrittura delle regole con un numero più basso. Per impostazione predefinita, l'ordine delle regole tra ingressi all'interno dello stesso gruppo di ingressi viene determinato in base all'ordine lessicografico dello spazio dei nomi e del nome.

> **Important**
>
> Assicurarsi che ogni ingresso nello stesso gruppo di ingressi disponga di un numero di priorità univoco. Non è possibile avere numeri d'ordine duplicati tra gli ingressi.

(Facoltativo) Implementare un'applicazione di esempio

Prerequisiti

- Almeno una sottorete pubblica o privata nel VPC del cluster.
- Aver implementato AWS Load Balancer Controller nel cluster. Per ulteriori informazioni, consulta Che cosa è la AWS Load Balancer Controller?. Consigliamo la versione `2.7.2` o successiva.

Per implementare un'applicazione di esempio

Puoi eseguire l'applicazione di esempio in un cluster che abbia nodi Amazon EC2, Pods Fargate o entrambi.

1. Se non stai eseguendo l'implementazione su Fargate, salta questa fase. Se si sta eseguendo l'implementazione su Fargate, creare un profilo Fargate. É possibile creare il profilo eseguendo il seguente comando o nella AWS Management Console utilizzando gli stessi valori per `name` e `namespace` che si trovano nel comando. Sostituisci i *`example values`* con i valori in tuo possesso.

   ```
   eksctl create fargateprofile \
       --cluster my-cluster \
       --region region-code \
       --name alb-sample-app \
   ```

```
    --namespace game-2048
```

2. Implementa il gioco 2048 come applicazione di esempio per verificare che AWS Load Balancer Controller crei un AWS ALB come risultato dell'oggetto di ingresso. Completare i passaggi per il tipo di sottorete verso la quale si sta eseguendo l'implementazione.

   a. Se stai eseguendo l'implementazione su Pods in un cluster creato con la famiglia `IPv6`, passa alla fase successiva.

      - Pubblica

        ```
        kubectl apply -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
        ```

      - Privata

        1. Eseguire il download del manifesto.

           ```
           curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
           ```

        2. Modificare il file e individuare la riga `alb.ingress.kubernetes.io/scheme: internet-facing`.
        3. Modificare *`internet-facing`* in **`internal`** e salvare il file.
        4. Applicare il file manifesto al cluster.

           ```
           kubectl apply -f 2048_full.yaml
           ```

   b. Se si sta eseguendo l'implementazione su Pods in un cluster creato con la famiglia `IPv6`, completa la seguente procedura.

      1. Eseguire il download del manifesto.

         ```
         curl -O https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
         ```

      2. Aprire il file in un editor e aggiungere la seguente riga alle annotazioni nella specifica di ingresso.

         ```
         alb.ingress.kubernetes.io/ip-address-type: dualstack
         ```

3. Se stai bilanciando il carico sui Pods interni piuttosto che sui Pods rivolti a Internet, modifica la riga che indica `alb.ingress.kubernetes.io/scheme: internet-facing` in `alb.ingress.kubernetes.io/scheme: internal`
4. Salvare il file.
5. Applicare il file manifesto al cluster.

```
kubectl apply -f 2048_full.yaml
```

3. Dopo pochi minuti, verificare che la risorsa di ingresso sia stata creata con il comando seguente.

```
kubectl get ingress/ingress-2048 -n game-2048
```

Di seguito viene riportato un output di esempio:

```
NAME           CLASS    HOSTS   ADDRESS
                       PORTS   AGE
ingress-2048   <none>   *       k8s-game2048-ingress2-xxxxxxxxxx-yyyyyyyyyy.region-
code.elb.amazonaws.com   80      2m32s
```

> **Note**
>
> Se hai creato il load balancer in una sottorete privata, il valore in ADDRESS nell'output precedente è preceduto da `internal-`.
> Se, dopo alcuni minuti, l'ingresso non è stato creato utilizzare il comando seguente per visualizzare i registri del AWS Load Balancer Controller. Questi registri possono contenere messaggi di errore che consentono di diagnosticare eventuali problemi relativi all'implementazione.
>
> ```
> kubectl logs -f -n kube-system -l app.kubernetes.io/instance=aws-load-
> balancer-controller
> ```

4. Se è stata eseguita l'implementazione a una sottorete pubblica, aprire un browser e accedere all'URL ADDRESS dall'output del comando precedente per visualizzare l'applicazione di esempio. Se non si visualizza nulla, aggiornare il browser e riprovare. Se è stata eseguita l'implementazione a una sottorete privata, devi visualizzare la pagina da un dispositivo all'interno del VPC, ad esempio un bastion host. Per ulteriori informazioni, consulta Host bastione Linux su AWS.

5. Dopo aver provato l'applicazione di esempio, eliminarla utilizzando uno dei comandi seguenti.

   - Se è stato applicato il manifesto anziché una copia scaricata, utilizzare il comando seguente.

     ```
     kubectl delete -f https://raw.githubusercontent.com/kubernetes-sigs/aws-load-balancer-controller/v2.7.2/docs/examples/2048/2048_full.yaml
     ```

   - Se è stato il manifesto scaricato e modificato, utilizzare il seguente comando.

     ```
     kubectl delete -f 2048_full.yaml
     ```

Limitazione degli indirizzi IP esterni assegnabili ai servizi

I servizi Kubernetes possono essere raggiunti dall'interno di un cluster tramite:

- Un indirizzo IP del cluster assegnato automaticamente da Kubernetes
- Qualsiasi indirizzo IP specificato per la proprietà `externalIPs` in una specifica di servizio. Gli indirizzi IP esterni non sono gestiti da Kubernetes e sono responsabilità dell'amministratore del cluster. Gli Indirizzi IP esterni specificati con `externalIPs` sono diversi dall'indirizzo IP esterno assegnato a un servizio di tipo `LoadBalancer` da un provider cloud.

Per ulteriori informazioni sui servizi Kubernetes, consulta Servizi nella documentazione di Kubernetes. È possibile limitare gli indirizzi IP che possono essere specificati per `externalIPs` in una specifica di servizio.

Per limitare gli indirizzi IP che possono essere specificati per **`externalIPs`** in una specifica di servizio

1. Implementazione di `cert-manager` per gestire i certificati webhook. Per ulteriori informazioni, consulta la documentazione `cert-manager`.

   ```
   kubectl apply -f https://github.com/jetstack/cert-manager/releases/download/v1.5.4/cert-manager.yaml
   ```

2. Verifica che i `cert-manager` Pods siano in esecuzione.

   ```
   kubectl get pods -n cert-manager
   ```

Di seguito viene riportato un output di esempio:

```
NAME                                       READY   STATUS    RESTARTS   AGE
cert-manager-58c8844bb8-nlx7q              1/1     Running   0          15s
cert-manager-cainjector-745768f6ff-696h5   1/1     Running   0          15s
cert-manager-webhook-67cc76975b-4v4nk      1/1     Running   0          14s
```

3. Esaminare i servizi esistenti per assicurarsi che a nessuno di essi siano assegnati indirizzi IP esterni non contenuti nel blocco CIDR per cui si desidera limitare gli indirizzi.

```
kubectl get services -A
```

Di seguito viene riportato un output di esempio:

```
NAMESPACE                   NAME                                    TYPE
   CLUSTER-IP       EXTERNAL-IP     PORT(S)         AGE
cert-manager                cert-manager                            ClusterIP
   10.100.102.137   <none>          9402/TCP        20m
cert-manager                cert-manager-webhook                    ClusterIP
   10.100.6.136     <none>          443/TCP         20m
default                     kubernetes                              ClusterIP
   10.100.0.1       <none>          443/TCP         2d1h
externalip-validation-system   externalip-validation-webhook-service   ClusterIP
   10.100.234.179   <none>          443/TCP         16s
kube-system                 kube-dns                                ClusterIP
   10.100.0.10      <none>          53/UDP,53/TCP   2d1h
my-namespace                my-service                              ClusterIP
   10.100.128.10    192.168.1.1     80/TCP          149m
```

Se uno qualsiasi dei valori costituiscono indirizzi IP che non si trovano all'interno del blocco per cui si desidera limitare l'accesso, sarà necessario modificare gli indirizzi in modo che si trovino all'interno del blocco e implementare nuovamente i servizi. Ad esempio, al servizio `my-service` nell'output precedente è assegnato un indirizzo IP esterno che non rientra nel blocco CIDR di esempio del passaggio 5.

4. Eseguire il download del manifesto del webhook IP esterno. Puoi visualizzare il codice sorgente per il webhook anche su GitHub.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/externalip-webhook.yaml
```

5. Specifica i blocchi CIDR. Aprire il file scaricato nell'editor e rimuovere il # all'inizio delle righe seguenti.

```
#args:
#- --allowed-external-ip-cidrs=10.0.0.0/8
```

Sostituire `10.0.0.0/8` con il proprio blocco CIDR. É possibile specificare tutti i blocchi che si desiderano. Se si specificano blocchi multipli, aggiungere una virgola tra i blocchi.

6. Se il cluster non è nella Regione AWS `us-west-2`, `us-west-2`, `602401143452` e `amazonaws.com` nel file con i comandi seguenti. Prima di eseguire i comandi, sostituisci *`region-code`* e *`111122223333`* con il valore per la tua Regione AWS dall'elenco in Registri delle immagini del container Amazon.

```
sed -i.bak -e 's|602401143452|111122223333|' externalip-webhook.yaml
sed -i.bak -e 's|us-west-2|region-code|' externalip-webhook.yaml
sed -i.bak -e 's|amazonaws.com||' externalip-webhook.yaml
```

7. Applica il manifesto al cluster.

```
kubectl apply -f externalip-webhook.yaml
```

Qualsiasi tentativo di implementare un servizio nel cluster con un indirizzo IP specificato per `externalIPs` che non è contenuto nei blocchi specificati nella fase Specifica i blocchi CIDR avrà esito negativo.

Copia di un'immagine di container da un repository a un altro

In questo argomento viene descritto come estrarre un'immagine di container da un repository a cui i nodi non hanno accesso e inviare l'immagine a un repository a cui i nodi hanno accesso. Puoi inviare l'immagine ad Amazon ECR o a un repository alternativo al quale i tuoi nodi hanno accesso.

Prerequisiti

- Il motore Docker installato e configurato sul computer. Per ulteriori informazioni, consulta Installazione del motore Docker nella documentazione di Docker.
- La versione `2.12.3` o successive o quella `1.27.160` o successive dell'AWS Command Line Interface (AWS CLI) installata e configurata sul dispositivo o in AWS CloudShell. Per verificare la

versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface. La versione della AWS CLI installata in AWS CloudShell potrebbe anche essere di diverse versioni precedenti alla più recente. Per aggiornarla, consulta la sezione Installazione della AWS CLI nella directory principale nella Guida per l'utente di AWS CloudShell.

- Un endpoint VPC di interfaccia per Amazon ECR se desideri che i tuoi nodi estraggano le immagini dei container o trasferiscano le immagini dei container in un repository Amazon ECR privato sulla rete di Amazon. Per ulteriori informazioni, consulta la sezione Creazione degli endpoint VPC per Amazon ECR nella Guida per l'utente di Amazon Elastic Container Registry.

Completa i passaggi seguenti per estrarre un'immagine del container da un repository e inviarla al tuo repository. Negli esempi riportati in questo argomento, viene estratta l'immagine per l'helper di argomenti del Amazon VPC CNI plugin for Kubernetes. Quando segui questi passaggi, assicurati di sostituire i *`example values`* con i tuoi valori.

Copia di un'immagine di container da un repository a un altro

1. Se non disponi ancora di un repository Amazon ECR o di un altro repository, creane uno a cui i tuoi nodi hanno accesso. Il seguente comando crea un repository privato Amazon ECR. Il nome di un repository privato Amazon ECR deve iniziare con una lettera. Può contenere soltanto lettere minuscole, numeri, trattini (-), trattini bassi (_) e barre (/). Per ulteriori informazioni, consulta la sezione Creazione di un repository privato nella Guida per l'utente di Amazon Elastic Container Registry.

   Puoi sostituire *`cni-metrics-helper`* con un qualsiasi valore a tua scelta. Come best practice, crea un repository separato per ogni immagine. Lo consigliamo perché i tag immagine devono essere univoci all'interno di un repository. Sostituisci *`region-code`* con una Regione AWS supportata da Amazon ECR.

   ```
   aws ecr create-repository --region region-code --repository-name cni-metrics-helper
   ```

2. Determina il registro, il repository e il tag (facoltativo) dell'immagine che i nodi devono estrarre. Questa informazione è nel formato `registry/repository[:tag]`.

Molti degli argomenti di Amazon EKS sull'installazione delle immagini richiedono l'applicazione di un file manifesto o l'installazione dell'immagine utilizzando un grafico Helm. Tuttavia, prima di applicare un file manifesto o installare un grafico Helm, visualizza il contenuto del manifesto o del file `values.yaml` del grafico. In questo modo, puoi determinare il registro, il repository e il tag da estrarre.

Ad esempio, puoi trovare la seguente riga nel file manifesto per l'helper di parametri del Amazon VPC CNI plugin for Kubernetes. Il registro è `602401143452.dkr.ecr.us-west-2.amazonaws.com`, che è un registro privato Amazon ECR. Il repository è `cni-metrics-helper`.

```
image: "602401143452.dkr.ecr.us-west-2.amazonaws.com/cni-metrics-helper:v1.12.6"
```

Potresti visualizzate le seguenti varianti per la posizione dell'immagine:

- Solo `repository-name:tag`. In questo caso, `docker.io` di solito è il registro, ma non è specificato poiché per impostazione predefinita Kubernetes gli antepone un nome di repository se non viene specificato alcun registro.
- `repository-name/repository-namespace/repository:tag`. Lo spazio dei nomi del repository è facoltativo, ma talvolta viene specificato dal proprietario del repository per la classificazione delle immagini. Ad esempio, tutte le immagini Amazon EC2 nella galleria pubblica Amazon ECR usano lo spazio dei nomi `aws-ec2`.

Prima di installare un'immagine con Helm, visualizza il file `values.yaml` Helm per determinare la posizione dell'immagine. Ad esempio, il file `values.yaml` per l'helper di parametri del Amazon VPC CNI plugin for Kubernetes include le seguenti righe.

```
image:
  region: us-west-2
  tag: v1.12.6
  account: "602401143452"
  domain: "amazonaws.com"
```

3. Estrai l'immagine del container specificata nel file manifesto.

   a. Se stai eseguendo l'estrazione da un registro pubblico, come ad esempio la Galleria pubblica di Amazon ECR, puoi passare alla fase secondaria successiva poiché l'autenticazione non

è richiesta. In questo esempio, esegui l'autenticazione a un registro privato di Amazon ECR contenente il repository per l'immagine dell'helper per i parametri CNI. Amazon EKS conserva l'immagine in ogni registro elencato nei Registri delle immagini del container Amazon. Puoi eseguire l'autenticazione a uno qualsiasi dei registri sostituendo *602401143452* e *region-code* con le informazioni di un registro differente. Esiste un registro separato per ciascuna Regione AWS in cui Amazon EKS è supportato.

```
aws ecr get-login-password --region region-code | docker login --username AWS --password-stdin 602401143452.dkr.ecr.region-code.amazonaws.com
```

b. Estrai l'immagine. In questo esempio, esegui l'estrazione dal registro a cui ti sei autenticato nel passaggio secondario precedente. Sostituisci *602401143452* e *region-code* con le informazioni fornite nel passaggio secondario precedente.

```
docker pull 602401143452.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

4. Tagga l'immagine che hai estratto con il registro, il repository e il tag. L'esempio seguente presuppone che tu abbia estratto l'immagine dal file manifesto e la invierai al repository privato Amazon ECR creato nel primo passaggio. Sostituisci *111122223333* con l'ID del tuo account. Sostituisci *region-code* con la Regione AWS in cui hai creato il repository privato di Amazon ECR.

```
docker tag cni-metrics-helper:v1.12.6 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

5. Esegui l'autenticazione al tuo registro. In questo esempio, esegui l'autenticazione al registro privato Amazon ECR creato nel primo passaggio. Per ulteriori informazioni, consulta la sezione Autenticazione del registro nella Guida per l'utente di Amazon Elastic Container Registry.

```
aws ecr get-login-password --region region-code | docker login --username AWS --password-stdin 111122223333.dkr.ecr.region-code.amazonaws.com
```

6. Invia l'immagine al tuo repository. In questo esempio, invii l'immagine al registro privato Amazon ECR creato nel primo passaggio. Per ulteriori informazioni, consulta Invio di un'immagine Docker nella Guida per l'utente di Amazon Elastic Container Registry.

```
docker push 111122223333.dkr.ecr.region-code.amazonaws.com/cni-metrics-helper:v1.12.6
```

7. Aggiorna il file manifesto utilizzato per determinare l'immagine in un passaggio precedente con il `registry/repository:tag` per l'immagine che hai inviato. Se stai eseguendo l'installazione con un grafico Helm, spesso hai la possibilità di specificare il `registry/repository:tag`. Quando installi il grafico, specifica il `registry/repository:tag` dell'immagine che hai inviato al repository.

Registri delle immagini del container Amazon

Quando implementi AWS Componenti aggiuntivi Amazon EKS nel tuo cluster, i nodi estraggono le immagini dei container necessarie dal registro specificato nel meccanismo di installazione del componente aggiuntivo, ad esempio un manifesto di installazione o un file `values.yaml` Helm. Le immagini vengono estratte da un repository privato Amazon ECR di Amazon EKS. Amazon EKS replica le immagini in un repository in ciascun Regione AWS supportato da Amazon EKS. I nodi possono estrarre l'immagine del container su Internet da uno dei seguenti registri. In alternativa, i tuoi nodi possono estrarre l'immagine sulla rete di Amazon se hai creato un'interfaccia endpoint VPC per Amazon ECR (AWS PrivateLink) nel VPC. I registri richiedono l'autenticazione con un account IAM AWS. I nodi si autenticano utilizzando il ruolo IAM del nodo Amazon EKS, dotato delle autorizzazioni nella policy IAM gestita da AmazonEC2ContainerRegistryReadOnly associata.

Regione AWS	Registro
af-south-1	877085696533.dkr.ecr.af-south-1.amazonaws.com
ap-east-1	800184023465.dkr.ecr.ap-east-1.amazonaws.com
ap-northeast-1	602401143452.dkr.ecr.ap-northeast-1.amazonaws.com
ap-northeast-2	602401143452.dkr.ecr.ap-northeast-2.amazonaws.com
ap-northeast-3	602401143452.dkr.ecr.ap-northeast-3.amazonaws.com

Regione AWS	Registro
ap-south-1	602401143452.dkr.ecr.ap-south-1.amazonaws.com
ap-south-2	900889452093.dkr.ecr.ap-south-2.amazonaws.com
ap-southeast-1	602401143452.dkr.ecr.ap-southeast-1.amazonaws.com
ap-southeast-2	602401143452.dkr.ecr.ap-southeast-2.amazonaws.com
ap-southeast-3	296578399912.dkr.ecr.ap-southeast-3.amazonaws.com
ap-southeast-4	491585149902.dkr.ecr.ap-southeast-4.amazonaws.com
ca-central-1	602401143452.dkr.ecr.ca-central-1.amazonaws.com
ca-west-1	761377655185.dkr.ecr.ca-west-1.amazonaws.com
cn-north-1	918309763551.dkr.ecr.cn-north-1.amazonaws.com.cn
cn-northwest-1	961992271922.dkr.ecr.cn-northwest-1.amazonaws.com.cn
eu-central-1	602401143452.dkr.ecr.eu-central-1.amazonaws.com
eu-central-2	900612956339.dkr.ecr.eu-central-2.amazonaws.com
eu-north-1	602401143452.dkr.ecr.eu-north-1.amazonaws.com

Regione AWS	Registro
eu-south-1	590381155156.dkr.ecr.eu-south-1.amazonaws.com
eu-south-2	455263428931.dkr.ecr.eu-south-2.amazonaws.com
eu-west-1	602401143452.dkr.ecr.eu-west-1.amazonaws.com
eu-west-2	602401143452.dkr.ecr.eu-west-2.amazonaws.com
eu-west-3	602401143452.dkr.ecr.eu-west-3.amazonaws.com
il-central-1	066635153087.dkr.ecr.il-central-1.amazonaws.com
me-south-1	558608220178.dkr.ecr.me-south-1.amazonaws.com
me-central-1	759879836304.dkr.ecr.me-central-1.amazonaws.com
sa-east-1	602401143452.dkr.ecr.sa-east-1.amazonaws.com
us-east-1	602401143452.dkr.ecr.us-east-1.amazonaws.com
us-east-2	602401143452.dkr.ecr.us-east-2.amazonaws.com
us-gov-east-1	151742754352.dkr.ecr.us-gov-east-1.amazonaws.com
us-gov-west-1	013241004608.dkr.ecr.us-gov-west-1.amazonaws.com

Regione AWS	Registro
us-west-1	602401143452.dkr.ecr.us-west-1.amazonaws.com
us-west-2	602401143452.dkr.ecr.us-west-2.amazonaws.com

Componenti aggiuntivi Amazon EKS

Un componente aggiuntivo è un software che fornisce funzionalità operative di supporto alle applicazioni Kubernetes, ma non è specifico per l'applicazione. Rientrano in questa categoria software come agenti di osservabilità o driver Kubernetes che consentono al cluster di interagire con risorse AWS sottostanti per la rete, il calcolo e l'archiviazione. Il software aggiuntivo viene in genere creato e gestito dalla Kubernetes community, dai provider di servizi cloud o da fornitori di terze parti. AWS Per ogni cluster, Amazon EKS installa in automatico componenti aggiuntivi autogestiti come Amazon VPC CNI plugin for Kubernetes, `kube-proxy` e CoreDNS. È possibile modificare la configurazione predefinita dei componenti aggiuntivi e aggiornarli quando lo desideri.

I componenti aggiuntivi di Amazon EKS forniscono l'installazione e la gestione di un set curato di componenti aggiuntivi per i cluster Amazon EKS. Tutti i componenti aggiuntivi di Amazon EKS includono le patch di sicurezza e le correzioni di bug più recenti e sono convalidati per AWS funzionare con Amazon EKS. I componenti aggiuntivi Amazon EKS permettono di garantire costantemente sicurezza e stabilità ai cluster Amazon EKS e di ridurre la quantità di lavoro necessaria per installare, configurare e aggiornare i componenti aggiuntivi. Se un componente aggiuntivo autogestito, ad esempio `kube-proxy` è già in esecuzione nel cluster ed è disponibile come componente aggiuntivo di Amazon EKS, è possibile installare il componente aggiuntivo `kube-proxy` Amazon EKS per iniziare a beneficiare delle funzionalità dei componenti aggiuntivi di Amazon EKS.

Attraverso l'API di Amazon EKS puoi aggiornare campi di configurazione specifici gestiti da Amazon EKS per i componenti aggiuntivi di Amazon EKS. É anche possibile modificare i campi di configurazione non gestiti da Amazon EKS direttamente all'interno del cluster Kubernetes una volta avviato il componente aggiuntivo. Ciò include, ove applicabile, la definizione di campi di configurazione specifici per un componente aggiuntivo. Una volta apportate, queste modifiche non vengono sovrascritte da Amazon EKS. Ciò è reso possibile utilizzando la caratteristica applicazione lato server Kubernetes. Per ulteriori informazioni, consulta Gestione dei campi Kubernetes.

È possibile utilizzare i componenti aggiuntivi di Amazon EKS con qualsiasi tipo di nodo Amazon EKS.

Considerazioni

- Per configurare i componenti aggiuntivi per il cluster, il principale IAM deve disporre delle autorizzazioni IAM per l'utilizzo dei componenti aggiuntivi. Per ulteriori informazioni, consulta le operazioni con Addon nel loro nome in Operazioni definite da Amazon Elastic Kubernetes Service.
- I componenti aggiuntivi di Amazon EKS vengono eseguiti sui nodi con provisioning o configurazione per il cluster. I tipi di nodi includono istanze Amazon EC2 e Fargate.
- Puoi modificare i campi non gestiti da Amazon EKS per personalizzare l'installazione di un componente aggiuntivo di Amazon EKS. Per ulteriori informazioni, consulta Gestione dei campi Kubernetes.
- Se crei un cluster con AWS Management Console, i componenti aggiuntivi Amazon `kube-proxy` EKS e CoreDNS Amazon EKS vengono aggiunti automaticamente al cluster. Amazon VPC CNI plugin for Kubernetes Se utilizzi `eksctl` per creare il cluster con un file `config`, `eksctl` può anche creare il cluster con i componenti aggiuntivi di Amazon EKS. Se crea il cluster utilizzando `eksctl` senza un file `config` o con qualsiasi altro strumento, il `kube-proxy` autogestito, Amazon VPC CNI plugin for Kubernetes e i componenti aggiuntivi di CoreDNS vengono installati in alternativa ai componenti aggiuntivi di Amazon EKS. Puoi gestirli in autonomia o aggiungere manualmente i componenti aggiuntivi di Amazon EKS dopo la creazione del cluster.
- `eks:addon-cluster-admin` `ClusterRoleBinding` associa `cluster-admin` `ClusterRole` all'identità `eks:addon-manager` Kubernetes. Il ruolo dispone delle autorizzazioni necessarie all'identità `eks:addon-manager` per la creazione di spazi dei nomi Kubernetes e l'installazione di componenti aggiuntivi negli spazi dei nomi. Se il `ClusterRoleBinding` `eks:addon-cluster-admin` viene rimosso, il cluster Amazon EKS continua a funzionare, ma Amazon EKS non è più in grado di gestire i componenti aggiuntivi. Tutti i cluster a partire dalle seguenti versioni della piattaforma utilizzano il nuovo `ClusterRoleBinding`.

Versione di Kubernetes	Versione della piattaforma Amazon EKS
1.28	eks.2012

Versione di Kubernetes supportata da Amazon EKS
1.21
1.22
1.23
1.24

Puoi aggiungere, aggiornare o eliminare componenti aggiuntivi Amazon EKS utilizzando l'API Amazon EKS, AWS Management Console AWS CLI, `eeksctl`. Per ulteriori informazioni, consulta Gestione dei componenti aggiuntivi di Amazon EKS. La creazione di componenti aggiuntivi di Amazon EKS è possibile anche utilizzando AWS CloudFormation.

Componenti aggiuntivi Amazon EKS disponibili da Amazon EKS

Nel tuo cluster puoi creare i seguenti componenti aggiuntivi di Amazon EKS. Puoi sempre visualizzare l'elenco più aggiornato dei componenti aggiuntivi disponibili utilizzando`eksctl`, il AWS Management Console, o il. AWS CLI Per visualizzare tutti i componenti aggiuntivi disponibili o per installare un componente aggiuntivo, consulta la pagina Creazione di un componente aggiuntivo. Se un componente aggiuntivo richiede autorizzazioni IAM, è necessario disporre di un provider OpenID Connect (OIDC) IAM per il cluster. Per stabilire se ne possiedi uno o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster. Dopo avere installato un componente aggiuntivo, lo puoi aggiornare o eliminare.

Scegli un componente aggiuntivo per avere maggiori informazioni sul componente e sui relativi requisiti di installazione.

Amazon VPC CNI plugin for Kubernetes

- Nome: `vpc-cni`

- Descrizione: un plugin container network interface (CNI) Kubernetes che fornisce una rete VPC nativa per il cluster. Il tipo autogestito o gestito di questo componente aggiuntivo è installato su ogni nodo Amazon EC2 per impostazione predefinita.
- Autorizzazioni IAM richieste: questo componente aggiuntivo utilizza la funzionalità Ruoli IAM per gli account di servizio di Amazon EKS. Se il cluster utilizza la famiglia `IPv4`, sono necessarie le autorizzazioni in AmazonEKS_CNI_Policy. Se il cluster utilizza la famiglia `IPv6`, è necessario creare una policy IAM con le autorizzazioni in modalità IPv6. Puoi creare un ruolo IAM, collegare una delle policy e annotare l'account del servizio Kubernetes utilizzato dal componente aggiuntivo con il comando seguente.

  Sostituisci *`my-cluster`* con il nome del cluster e *`AmazonEKSVPCCNIRole`* con il nome del ruolo. Se il tuo cluster utilizza la famiglia `IPv6`, sostituisci *`AmazonEKS_CNI_Policy`* con il nome della policy che hai creato. Questo comando richiede che `eksctl` sia installato sul tuo dispositivo. Se devi utilizzare uno strumento diverso per creare il ruolo, collegarvi la policy e prendere nota dell'account del servizio Kubernetes, consulta la pagina Configurare un account Kubernetes di servizio per assumere un ruolo IAM.

  ```
  eksctl create iamserviceaccount --name aws-node --namespace kube-system --cluster my-cluster --role-name AmazonEKSVPCCNIRole \
      --role-only --attach-policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy --approve
  ```

- Informazioni aggiuntive: per ulteriori informazioni sulle impostazioni configurabili del componente aggiuntivo, consulta aws-vpc-cni-k8s su. GitHub Per ulteriori informazioni sul plug-in, consulta Proposta: plug-in CNI per il Kubernetes networking tramite AWS VPC. Per ulteriori informazioni sulla creazione di un componente aggiuntivo, consulta Creare il componente aggiuntivo di Amazon EKS.
- Informazioni sull'aggiornamento: puoi aggiornare solo una versione secondaria alla volta. Ad esempio, se la versione corrente è `1.28.`*`x`*`-eksbuild.`*`y`* e desideri aggiornarla a `1.30.`*`x`*`-eksbuild.`*`y`*, devi prima eseguire l'aggiornamento a `1.29.`*`x`*`-eksbuild.`*`y`* e quindi l'aggiornamento a `1.30.`*`x`*`-eksbuild.`*`y`*. Per ulteriori informazioni sull'aggiornamento del componente aggiuntivo, consulta Aggiornamento del componente aggiuntivo di Amazon EKS.

CoreDNS

- Nome: `coredns`

- Descrizione: è un server DNS flessibile ed estensibile che può fungere da DNS del cluster Kubernetes. Il tipo autogestito o gestito di questo componente aggiuntivo è stato installato, per impostazione predefinita, al momento della creazione del cluster. Quando si avvia un cluster Amazon EKS con almeno un nodo, due repliche dell'immagine CoreDNS vengono implementate per impostazione predefinita, a prescindere dal numero di nodi implementati nel cluster. I Pods CoreDNS forniscono la risoluzione dei nomi per tutti i Pods del cluster. Puoi implementare i Pods CoreDNS sui nodi Fargate se il cluster include un AWS Fargate profilo con uno spazio dei nomi che corrisponde allo spazio per `deployment` CoreDNS.
- Autorizzazioni IAM richieste: questo componente aggiuntivo non richiede alcuna autorizzazione.
- Informazioni aggiuntive: per ulteriori informazioni su CoreDNS, consulta le pagine Using CoreDNS for Service Discovery (Utilizzo di CoreDNS per il rilevamento dei servizi) e Customizing DNS Service (Personalizzazione del servizio DNS) nella documentazione di Kubernetes.

`Kube-proxy`

- Nome: `kube-proxy`
- Descrizione: mantiene le regole di rete su ogni nodo Amazon EC2. Consente la comunicazione di rete con i tuoi Pods. Il tipo autogestito o gestito di questo componente aggiuntivo è installato su ogni nodo Amazon EC2 nel cluster per impostazione predefinita.
- Autorizzazioni IAM richieste: questo componente aggiuntivo non richiede alcuna autorizzazione.
- Informazioni aggiuntive: per ulteriori informazioni`kube-proxy`, consulta la sezione `kube-proxy` nella documentazione di Kubernetes.
- Informazioni sull'aggiornamento: prima di aggiornare la versione corrente, considera i seguenti requisiti:
  - `Kube-proxy` su un cluster Amazon EKS ha la stessa policy di compatibilità e disallineamento di Kubernetes.
  - `Kube-proxy` deve essere alla stessa versione secondaria di `kubelet` sui nodi Amazon EC2.
  - `Kube-proxy` non può essere successivo alla versione secondaria del piano di controllo (control plane) del tuo cluster.
  - La versione `kube-proxy` sui nodi Amazon EC2 non può essere di più di due versioni secondarie successiva rispetto al piano di controllo. Ad esempio, se il tuo piano di controllo è in esecuzione alla Kubernetes 1.30, la versione `kube-proxy` secondaria non può essere precedente alla 1.28.

- Se hai aggiornato recentemente il cluster a una nuova versione secondaria di Kubernetes, aggiorna i nodi EC2 Amazon alla medesima versione secondaria prima di eseguire l'aggiornamento di `kube-proxy` alla stessa versione secondaria dei nodi.

Driver CSI per Amazon EBS

- Nome: `aws-ebs-csi-driver`
- Descrizione: un plugin Container Storage Interface (CSI) Kubernetes che fornisce archiviazione Amazon EBS per il cluster.
- Autorizzazioni IAM richieste: questo componente aggiuntivo utilizza la funzionalità Ruoli IAM per gli account di servizio di Amazon EKS. Le autorizzazioni nella policy `AmazonEBSCSIDriverPolicy` AWS gestita sono obbligatorie. Puoi creare un ruolo IAM e collegarvi la policy gestita con il seguente comando. Sostituisci *`my-cluster`* con il nome del cluster e *`AmazonEKS_EBS_CSI_DriverRole`* con il nome del ruolo. Questo comando richiede che `eksctl` sia installato sul tuo dispositivo. Se hai la necessità di utilizzare uno strumento diverso o una chiave KMS personalizzata per la crittografia, consulta la sezione Creazione del ruolo IAM per il driver CSI per Amazon EBS.

```
eksctl create iamserviceaccount \
    --name ebs-csi-controller-sa \
    --namespace kube-system \
    --cluster my-cluster \
    --role-name AmazonEKS_EBS_CSI_DriverRole \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy
 \
    --approve
```

- Informazioni aggiuntive: per ulteriori informazioni sul componente aggiuntivo, consulta Driver CSI per Amazon EBS.

Driver CSI per Amazon EFS

- Nome: `aws-efs-csi-driver`
- Descrizione: un plugin Container Storage Interface (CSI) Kubernetes che fornisce archiviazione Amazon EFS per il cluster.

- Autorizzazioni IAM richieste: questo componente aggiuntivo utilizza la funzionalità Ruoli IAM per gli account di servizio di Amazon EKS. Le autorizzazioni nella politica `AmazonEFSCSIDriverPolicy` AWS gestita sono obbligatorie. Puoi creare un ruolo IAM e collegarvi la policy gestita con i seguenti comandi. Sostituisci *`my-cluster`* con il nome del cluster e *`AmazonEKS_EFS_CSI_DriverRole`* con il nome del ruolo. Questi comandi richiedono l'installazione di `eksctl` sul tuo dispositivo. Se hai bisogno di usare uno strumento diverso, vedi Creazione di un ruolo IAM.

```
export cluster_name=my-cluster
export role_name=AmazonEKS_EFS_CSI_DriverRole
eksctl create iamserviceaccount \
    --name efs-csi-controller-sa \
    --namespace kube-system \
    --cluster $cluster_name \
    --role-name $role_name \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/service-role/AmazonEFSCSIDriverPolicy
 \
    --approve
TRUST_POLICY=$(aws iam get-role --role-name $role_name --query
 'Role.AssumeRolePolicyDocument' | \
    sed -e 's/efs-csi-controller-sa/efs-csi-*/' -e 's/StringEquals/StringLike/')
aws iam update-assume-role-policy --role-name $role_name --policy-document
 "$TRUST_POLICY"
```

- Informazioni aggiuntive: per ulteriori informazioni sul componente aggiuntivo, consulta Driver CSI per Amazon EFS.

Driver CSI di Mountpoint per Amazon S3

- Nome: `aws-mountpoint-s3-csi-driver`
- Descrizione: un plugin CSI (Container Storage Interface) di Kubernetes che fornisce l'archiviazione Amazon S3 per il cluster.
- Autorizzazioni IAM richieste: questo componente aggiuntivo utilizza la funzionalità Ruoli IAM per gli account di servizio di Amazon EKS. Il ruolo IAM creato richiederà una policy che fornisca l'accesso a S3. Segui i Suggerimenti sulle autorizzazioni Mountpoint IAM durante la creazione della policy. In alternativa, è possibile utilizzare la politica AWS gestita AmazonS3FullAccess, ma questa politica gestita concede più autorizzazioni di quelle per cui sono necessarie. Mountpoint

Puoi creare un ruolo IAM e collegarvi la policy gestita con i seguenti comandi. *Sostituisci my-cluster con il nome del tuo cluster, il codice regionale con il Regione AWS codice corretto, AmazoneKS_S3_CSI_ con il nome del tuo ruolo e DriverRoleAmazoneKS_S3_CSI_ _ARN con il ruolo ARN. DriverRole* Questi comandi richiedono l'installazione di `eksctl` sul tuo dispositivo. Per istruzioni sull'Creazione di un ruolo IAMuso della console IAM oppure, consulta. AWS CLI

```
CLUSTER_NAME=my-cluster
REGION=region-code
ROLE_NAME=AmazonEKS_S3_CSI_DriverRole
POLICY_ARN=AmazonEKS_S3_CSI_DriverRole_ARN
eksctl create iamserviceaccount \
    --name s3-csi-driver-sa \
    --namespace kube-system \
    --cluster $CLUSTER_NAME \
    --attach-policy-arn $POLICY_ARN \
    --approve \
    --role-name $ROLE_NAME \
    --region $REGION \
    --role-only
```

- Informazioni aggiuntive: per ulteriori informazioni sul componente aggiuntivo, consulta Driver CSI di Mountpoint per Amazon S3.

Controller di snapshot CSI

- Nome: `snapshot-controller`
- Descrizione: il controller di snapshot CSI (Container Storage Interface) consente l'uso della funzionalità di snapshot nei driver CSI compatibili, come il driver CSI di Amazon EBS.
- Autorizzazioni IAM richieste: questo componente aggiuntivo non richiede alcuna autorizzazione.
- Informazioni aggiuntive: per ulteriori informazioni sul componente aggiuntivo, consulta Controller di snapshot CSI.

AWS Distro per OpenTelemetry

- Nome: `adot`

- Descrizione: The AWS Distro for OpenTelemetry (ADOT) è una distribuzione del progetto sicura, pronta per la produzione e AWS supportata. OpenTelemetry
- Autorizzazioni IAM richieste: questo componente aggiuntivo richiede le autorizzazioni IAM solo se si utilizza una delle risorse personalizzate preconfigurate a cui è possibile accedere tramite la configurazione avanzata.
- Informazioni aggiuntive: per ulteriori informazioni, consulta la sezione Guida introduttiva a AWS Distro per l'OpenTelemetryutilizzo dei componenti aggiuntivi EKS nella distribuzione per la documentazione. AWS OpenTelemetry

  ADOT richiede che `cert-manager` sia implementato sul cluster come prerequisito, altrimenti questo componente aggiuntivo non funzionerà se implementato direttamente utilizzando la proprietà `cluster_addons` di Amazon EKS Terraform. Per ulteriori requisiti, consulta Requisiti per iniziare a usare AWS Distro per OpenTelemetry l'utilizzo dei componenti aggiuntivi EKS nella distribuzione per la documentazione. AWS OpenTelemetry

Amazon GuardDuty agente

- Nome: `aws-guardduty-agent`
- Descrizione: Amazon GuardDuty è un servizio di monitoraggio della sicurezza che analizza ed elabora fonti di dati fondamentali, inclusi eventi di AWS CloudTrail gestione e log di flusso Amazon VPC. Amazon elabora GuardDuty anche funzionalità, come i registri Kubernetes di controllo e il monitoraggio del runtime.
- Autorizzazioni IAM richieste: questo componente aggiuntivo non richiede alcuna autorizzazione.
- Informazioni aggiuntive: per ulteriori informazioni, consulta Runtime Monitoring for Amazon EKS clusters in Amazon GuardDuty.
  - Per rilevare potenziali minacce alla sicurezza nei tuoi cluster Amazon EKS, abilita il monitoraggio del GuardDuty runtime di Amazon e distribuisci l'agente GuardDuty di sicurezza nei tuoi cluster Amazon EKS.

Agente Amazon CloudWatch Observability

- Nome: `amazon-cloudwatch-observability`
- Descrizione Amazon CloudWatch Agent è il servizio di monitoraggio e osservabilità fornito da AWS. Questo componente aggiuntivo installa l' CloudWatch agente e abilita sia CloudWatch

Application Signals che CloudWatch Container Insights con una migliore osservabilità per Amazon EKS.

- Autorizzazioni IAM richieste: questo componente aggiuntivo utilizza la funzionalità Ruoli IAM per gli account di servizio di Amazon EKS. Le autorizzazioni contenute nelle CloudWatchAgentServerpolicy AWS gestite da AWSXrayWriteOnlyAccessand Policy sono obbligatorie. Puoi creare un ruolo IAM, collegare una delle policy gestite e annotare l'account del servizio Kubernetes utilizzato dal componente aggiuntivo con il comando seguente. Sostituisci *my-cluster* con il nome del cluster e *AmazonEKS_Observability_role* con il nome del ruolo. Questo comando richiede che `eksctl` sia installato sul tuo dispositivo. Se devi utilizzare uno strumento diverso per creare il ruolo, collegarvi la policy e prendere nota dell'account del servizio Kubernetes, consulta la pagina Configurare un account Kubernetes di servizio per assumere un ruolo IAM.

```
eksctl create iamserviceaccount \
    --name cloudwatch-agent \
    --namespace amazon-cloudwatch \
    --cluster my-cluster \
    --role-name AmazonEKS_Observability_Role \
    --role-only \
    --attach-policy-arn arn:aws:iam::aws:policy/AWSXrayWriteOnlyAccess \
    --attach-policy-arn arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy \
    --approve
```

- Informazioni aggiuntive: per ulteriori informazioni, consulta Installare l' CloudWatch agente.

Amazon EKS Pod Identity Agent

- Nome: `eks-pod-identity-agent`
- Descrizione: Amazon EKS Pod Identity offre la possibilità di gestire le credenziali per le applicazioni, in modo simile al modo in cui i profili di Amazon EC2 istanza forniscono le credenziali alle istanze EC2.
- Autorizzazioni IAM richieste: questo componente aggiuntivo utilizza le autorizzazioni di Ruolo IAM del nodo Amazon EKS.
- Informazioni sull'aggiornamento: puoi aggiornare solo una versione secondaria alla volta. Ad esempio, se la versione corrente è `1.28.`*x*`-eksbuild.`*y* e desideri aggiornarla a `1.30.`*x*`-eksbuild.`*y*, devi prima eseguire l'aggiornamento a `1.29.`*x*`-eksbuild.`*y* e quindi

l'aggiornamento a `1.30.x-eksbuild.y`. Per ulteriori informazioni sull'aggiornamento del componente aggiuntivo, consulta Aggiornamento del componente aggiuntivo di Amazon EKS.

Componenti aggiuntivi di Amazon EKS da fornitori di software indipendenti

Oltre all'elenco precedente di componenti aggiuntivi di Amazon EKS, puoi anche aggiungere un'ampia selezione di componenti software aggiuntivi operativi per Amazon EKS offerti da fornitori di software indipendenti. Scegli un componente aggiuntivo per avere maggiori informazioni sul componente e sui relativi requisiti di installazione.

Trova, acquista e distribuisci componenti aggiuntivi da AWS Marketplace ad Amazon EKS (). YouTube

Accuknox

- Editore: Accuknox
- Nome: `accuknox_kubearmor`
- Spazio dei nomi: `kubearmor`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta la sezione Guida introduttiva KubeArmor nella KubeArmor documentazione.

Akuity

- Editore: Akuity
- Nome: `akuity_agent`
- Spazio dei nomi: `akuity`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.

- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installazione dell'agente Akuity su Amazon EKS con il componente aggiuntivo Akuity EKS nella documentazione della piattaforma Akuity.

Calyptia

- Editore: Calyptia
- Nome: `calyptia_fluent-bit`
- Spazio dei nomi: `calytia-fluentbit`
- Nome dell'account di servizio: `clyptia-fluentbit`
- AWS politica IAM gestita AWSMarketplaceMeteringRegisterUsage—.
- Comando per creare il ruolo IAM richiesto – Il comando seguente richiede che tu disponga di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster. Sostituisci *`my-cluster`* con il nome del cluster e *`my-calyptia-role`* con il nome del ruolo. Questo comando richiede che l'installazione di `eksctl` sul tuo dispositivo. Se devi utilizzare uno strumento diverso per creare il ruolo e prendere nota dell'account di servizio Kubernetes, consulta Configurare un account Kubernetes di servizio per assumere un ruolo IAM.

  ```
  eksctl create iamserviceaccount --name service-account-name  --namespace calyptia-
  fluentbit --cluster my-cluster --role-name my-calyptia-role \
      --role-only --attach-policy-arn arn:aws:iam::aws:policy/
  AWSMarketplaceMeteringRegisterUsage --approve
  ```

- Istruzioni di configurazione e utilizzo: consulta Calyptia for Fluent Bit nella documentazione di Calyptia.

Cisco Observability Collector

- Editore: Cisco
- Nome: `cisco_cisco-cloud-observability-collectors`
- Spazio dei nomi: `appdynamics`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.

- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Utilizzare i componenti aggiuntivi Cisco Cloud Observability AWS Marketplace nella documentazione Cisco. AppDynamics

Cisco Observability Operator

- Editore: Cisco
- Nome: `cisco_cisco-cloud-observability-operators`
- Spazio dei nomi: `appdynamics`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Utilizzare i componenti aggiuntivi Cisco Cloud Observability AWS Marketplace nella documentazione Cisco. AppDynamics

CLOUDSOFT

- Editore: CLOUDSOFT
- Nome: `cloudsoft_cloudsoft-amp`
- Spazio dei nomi: `cloudsoft-amp`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Amazon EKS ADDON nella documentazione CLOUDSOFT.

Cribl

- Editore: Cribl
- Nome: `cribl_cribledge`
- Spazio dei nomi: `cribledge`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installazione del componente aggiuntivo Cribl di Amazon EKS per Edge nella documentazione di Cribl.

Dynatrace

- Editore: Dynatrace
- Nome: `dynatrace_dynatrace-operator`
- Spazio dei nomi: `dynatrace`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Custom IAM permissions (Autorizzazioni IAM personalizzate): con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni per la configurazione e l'uso: consulta Monitoraggio di Kubernetes nella documentazione di dynatrace.

Datree

- Autore – Datree
- Nome: `datree_engine-pro`
- Spazio dei nomi: `datree`
- Nome dell'account di servizio: datree-webhook-server-awsmp
- AWS policy IAM gestita — AWSLicenseManagerConsumptionPolicy.

- Comando per creare il ruolo IAM richiesto – Il comando seguente richiede che tu disponga di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster. Sostituisci *my-cluster* con il nome del cluster e *my-datree-role* con il nome del ruolo. Questo comando richiede che l'installazione di `eksctl` sul tuo dispositivo. Se devi utilizzare uno strumento diverso per creare il ruolo e prendere nota dell'account di servizio Kubernetes, consulta Configurare un account Kubernetes di servizio per assumere un ruolo IAM.

  ```
  eksctl create iamserviceaccount --name datree-webhook-server-awsmp --namespace datree
   --cluster my-cluster --role-name my-datree-role \
      --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
  AWSLicenseManagerConsumptionPolicy --approve
  ```

- Autorizzazioni IAM personalizzate – Con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni per la configurazione e l'utilizzo: consulta la sezione Integrazione di Amazon EKS nella documentazione di Datree.

Datadog

- Editore: Datadog
- Nome: `datadog_operator`
- Spazio dei nomi: `datadog-agent`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installazione dell'agente Datadog su Amazon EKS con il componente aggiuntivo Datadog Operator nella documentazione di Datadog.

Groundcover

- Editore: groundcover
- Nome: `groundcover_agent`
- Spazio dei nomi: `groundcover`

- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installazione del componente aggiuntivo groundcover di Amazon EKS nella documentazione di groundcover.

Grafana Labs

- Editore: Grafana Labs
- Nome: `grafana-labs_kubernetes-monitoring`
- Spazio dei nomi: `monitoring`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Configure Kubernetes Monitoring as an Add-on with Amazon EKS nella documentazione di Grafana Labs.

HA Proxy

- Editore: HA Proxy
- Nome: `haproxy-technologies_kubernetes-ingress-ee`
- Spazio dei nomi: `haproxy-controller`
- Nome dell'account di servizio: `customer defined`
- AWS policy IAM gestita — AWSLicenseManagerConsumptionPolicy.
- Comando per creare il ruolo IAM richiesto – Il comando seguente richiede che tu disponga di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster. Sostituisci *`my-cluster`* con il nome del cluster e *`my-haproxy-role`* con il nome del ruolo. Questo comando richiede che l'installazione di `eksctl` sul tuo dispositivo. Se devi utilizzare uno strumento diverso per creare

il ruolo e prendere nota dell'account di servizio Kubernetes, consulta Configurare un account Kubernetes di servizio per assumere un ruolo IAM.

```
eksctl create iamserviceaccount --name service-account-name  --namespace haproxy-
controller --cluster my-cluster --role-name my-haproxy-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- Autorizzazioni IAM personalizzate – Con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni per la configurazione e l'uso— VediInstalla HAProxy Enterprise Kubernetes Controller di ingresso su Amazon EKS di AWSnella documentazione di HAProxy.

Kpow

- Editore: Factorhouse
- Nome: `factorhouse_kpow`
- Spazio dei nomi: `factorhouse`
- Nome dell'account di servizio: `kpow`
- AWS politica IAM gestita — AWSLicenseManagerConsumptionPolicy
- Comando per creare il ruolo IAM richiesto – Il comando seguente richiede che tu disponga di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster. Sostituisci *`my-cluster`* con il nome del cluster e *`my-kpow-role`* con il nome del ruolo. Questo comando richiede che l'installazione di `eksctl` sul tuo dispositivo. Se devi utilizzare uno strumento diverso per creare il ruolo e prendere nota dell'account di servizio Kubernetes, consulta Configurare un account Kubernetes di servizio per assumere un ruolo IAM.

```
eksctl create iamserviceaccount --name kpow --namespace factorhouse --cluster my-
cluster --role-name my-kpow-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- Autorizzazioni IAM personalizzate – Con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni per la configurazione e l'uso: consulta Marketplace AWS LM nella documentazione di Kpow.

Kubecost

- Autore – Kubecost
- Nome: `kubecost_kubecost`
- Spazio dei nomi: `kubecost`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta AWS Cloud Billing Integration nella Kubecost documentazione.
- Se la versione del tuo cluster è `1.23` o successiva, nel tuo cluster deve essere installato the section called “Driver CSI per Amazon EBS”.

Kasten

- Autore – Kasten by Veeam
- Nome: `kasten_k10`
- Spazio dei nomi: `kasten-io`
- Nome dell'account di servizio: `k10-k10`
- AWS politica IAM gestita — AWSLicenseManagerConsumptionPolicy.
- Comando per creare il ruolo IAM richiesto – Il comando seguente richiede che tu disponga di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster. Sostituisci *`my-cluster`* con il nome del cluster e *`my-kasten-role`* con il nome del ruolo. Questo comando richiede che l'installazione di `eksctl` sul tuo dispositivo. Se devi utilizzare uno strumento diverso per creare il ruolo e prendere nota dell'account di servizio Kubernetes, consulta Configurare un account Kubernetes di servizio per assumere un ruolo IAM.

```
eksctl create iamserviceaccount --name k10-k10 --namespace kasten-io --cluster my-
cluster --role-name my-kasten-role \
    --role-only --attach-policy-arn arn:aws:iam::aws:policy/service-role/
AWSLicenseManagerConsumptionPolicy --approve
```

- Autorizzazioni IAM personalizzate – Con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installazione di K10 sull' AWS utilizzo del componente aggiuntivo Amazon EKS nella documentazione di Kasten.
- Informazioni aggiuntive: se il tuo cluster Amazon EKS è Kubernetes versione `1.23` o successiva, il driver Amazon EBS CSI deve essere installato sul cluster con una `StorageClass` predefinita.

Kong

- Editore: Kong
- Nome: `kong_konnect-ri`
- Spazio dei nomi: `kong`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installazione del componente aggiuntivo Kong Gateway di EKS nella documentazione di Kong.

LeakSignal

- Editore: LeakSignal
- Nome: `leaksignal_leakagent`
- Spazio dei nomi: `leakagent`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installare il LeakAgent componente aggiuntivo nella LeakSignal documentazione.

NetApp

- Editore: NetApp
- Nome: `netapp_trident-operator`
- Spazio dei nomi: `trident`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Configurare il componente aggiuntivo Astra Trident EKS nella documentazione. NetApp

New Relic

- Editore: New Relic
- Nome: `new-relic_kubernetes-operator`
- Spazio dei nomi: `newrelic`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installazione del componente aggiuntivo New Relic per EKS nella documentazione di New Relic.

Rafay

- Editore: Rafay
- Nome: `rafay-systems_rafay-operator`
- Spazio dei nomi: `rafay-system`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.

- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Installazione del componente aggiuntivo Rafay di Amazon EKS nella documentazione di Rafay.

Solo.io

- Editore: Solo.io
- Nome: `solo-io_istio-distro`
- Spazio dei nomi: `istio-system`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: vedi Installazione di Istio nella documentazione di Solo.io.

Stormforge

- Editore: Stormforge
- Nome: `stormforge_optimize-Live`
- Spazio dei nomi: `stormforge-system`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: vedi Installazione dell' StormForge agente nella StormForge documentazione.

Splunk

- Editore: Splunk

- Nome: `splunk_splunk-otel-collector-chart`
- Spazio dei nomi: `splunk-monitoring`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Install the Splunk add-on for Amazon EKS nella documentazione di Splunk.

Teleport

- Editore: Teleport
- Nome: `teleport_teleport`
- Spazio dei nomi: `teleport`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate – Con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Come funziona Teleport nella documentazione di Teleport.

Tetrate

- Editore: Tetrate Io
- Nome: `tetrate-io_istio-distro`
- Spazio dei nomi: `istio-system`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate – Con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.

- Istruzioni per la configurazione e l'uso: consulta il sito web Tetrate Istio Distro.

Upbound Universal Crossplane

- Autore – Upbound
- Nome: `upbound_universal-crossplane`
- Spazio dei nomi: `upbound-system`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate – Con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta Upbound Universal Crossplane (UXP) nella documentazione di Upbound.

Upwind

- Editore: Upwind
- Nome: `upwind`
- Spazio dei nomi: `upwind`
- Nome account di servizio – Con questo componente aggiuntivo non viene utilizzato un account di servizio.
- AWS policy IAM gestita: con questo componente aggiuntivo non viene utilizzata una policy gestita.
- Autorizzazioni IAM personalizzate: con questo componente aggiuntivo non vengono utilizzate autorizzazioni personalizzate.
- Istruzioni di configurazione e utilizzo: consulta i passaggi di installazione nella documentazione di Upwind.

Gestione dei componenti aggiuntivi di Amazon EKS

I componenti aggiuntivi di Amazon EKS sono un set selezionato di software aggiuntivi per cluster Amazon EKS. Tutti i componenti aggiuntivi di Amazon EKS:

- includono le patch di sicurezza e le correzioni dei bug più recenti.

- sono convalidati AWS per funzionare con Amazon EKS.
- riducono la quantità di lavoro necessaria per gestire il software del componente aggiuntivo.

Ti AWS Management Console avvisa quando è disponibile una nuova versione per un componente aggiuntivo Amazon EKS. Avviando semplicemente l'aggiornamento, Amazon EKS aggiorna automaticamente il software del componente aggiuntivo.

Per un elenco dei componenti aggiuntivi disponibili, consulta Componenti aggiuntivi Amazon EKS disponibili da Amazon EKS. Per ulteriori informazioni sulla gestione dei campi di Kubernetes, consulta Gestione dei campi Kubernetes.

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS.

Creazione di un componente aggiuntivo

Puoi creare un componente aggiuntivo Amazon EKS utilizzando`eksctl`, il AWS Management Console, o il AWS CLI. Se il componente aggiuntivo richiede un ruolo IAM, consulta i dettagli del componente aggiuntivo specifico alla pagina Componenti aggiuntivi Amazon EKS disponibili da Amazon EKS per conoscere i dettagli sulla creazione del ruolo.

eksctl

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Come creare un componente aggiuntivo di Amazon EKS utilizzando **`eksctl`**

1. Visualizza i nomi dei componenti aggiuntivi disponibili per una versione del cluster. Sostituisci *`1.30`* con la versione del cluster.

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 | grep AddonName
```

Di seguito viene riportato un output di esempio:

```
"AddonName": "aws-ebs-csi-driver",
                        "AddonName": "coredns",
                        "AddonName": "kube-proxy",
                        "AddonName": "vpc-cni",
                        "AddonName": "adot",
                        "AddonName": "dynatrace_dynatrace-operator",
                        "AddonName": "upbound_universal-crossplane",
                        "AddonName": "teleport_teleport",
                        "AddonName": "factorhouse_kpow",
                        [...]
```

2. Visualizza le versioni disponibili per il componente aggiuntivo da creare. Sostituisci *1.30* con la versione del cluster. Sostituisci *name-of-addon* con il nome del componente aggiuntivo di cui visualizzare le versioni. Il nome deve essere uno dei nomi restituiti nei passaggi precedenti.

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 --name name-of-addon | grep AddonVersion
```

L'output seguente è un esempio di ciò che viene restituito per il componente aggiuntivo denominato `vpc-cni`. Come vedi, il componente aggiuntivo ha diverse versioni disponibili.

```
"AddonVersions": [
    "AddonVersion": "v1.12.0-eksbuild.1",
    "AddonVersion": "v1.11.4-eksbuild.1",
    "AddonVersion": "v1.10.4-eksbuild.1",
    "AddonVersion": "v1.9.3-eksbuild.1",
```

3. Stabilisci se il componente aggiuntivo da creare è Amazon EKS o un componente aggiuntivo Marketplace AWS . Marketplace AWS Dispone di componenti aggiuntivi di terze parti che richiedono il completamento di passaggi aggiuntivi per creare il componente aggiuntivo.

```
eksctl utils describe-addon-versions --kubernetes-version 1.30 --name name-of-addon | grep ProductUrl
```

Se non viene restituito alcun output, il componente aggiuntivo è Amazon EKS. Se viene restituito un output, il componente aggiuntivo è un componente aggiuntivo. Marketplace AWS Il seguente output viene fornito per un componente aggiuntivo denominato `teleport_teleport`.

```
"ProductUrl": "https://aws.amazon.com/marketplace/pp?
sku=3bda70bb-566f-4976-806c-f96faef18b26"
```

Puoi trovare ulteriori informazioni sul componente aggiuntivo nella sezione Marketplace AWS Con l'URL restituito. Se il componente aggiuntivo richiede un abbonamento, puoi abbonarti al componente aggiuntivo attraverso Marketplace AWS. Se intendi creare un componente aggiuntivo da Marketplace AWS, il responsabile IAM che stai utilizzando per creare il componente aggiuntivo deve avere l'autorizzazione per creare il ruolo collegato al AWSServiceRoleForAWSLicenseManagerRoleservizio. Per ulteriori informazioni sull'assegnazione di autorizzazioni a un'entità IAM, consulta Aggiunta e rimozione di autorizzazioni per identità IAM nella Guida per l'utente di IAM.

4. Crea un componente aggiuntivo di Amazon EKS. Copia il comando seguente sul tuo dispositivo. Apportare le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato:

   - Sostituisci *my-cluster* con il nome del cluster.
   - Sostituisci *name-of-addon* con il nome del componente aggiuntivo da creare.
   - Se desideri una versione del componente aggiuntivo precedente all'ultima versione, sostituisci *latest* con il numero di versione da utilizzare restituito nell'output di un passaggio precedente.
   - Se il componente aggiuntivo utilizza un ruolo dell'account di servizio, sostituisci *111122223333* con l'ID dell'account e sostituisci *role-name* con il nome del ruolo. Per istruzioni sulla creazione di un ruolo per il tuo account di servizio, consulta la documentazione del componente aggiuntivo che stai creando. Per specificare un ruolo per l'account di servizio è necessario disporre di un provider IAM OpenID Connect (OIDC) per il cluster. Per stabilire se ne possiedi uno per il tuo cluster o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.

     Se il componente aggiuntivo non utilizza un ruolo per l'account di servizio, elimina ***--service-account-role-arn*** **arn:aws:iam::*111122223333*:role/*role-name***.
   - Questo comando di esempio sovrascrive la configurazione di qualunque versione autogestita esistente del componente aggiuntivo, se presente. Se non desideri sovrascrivere la configurazione di un componente aggiuntivo autogestito esistente, rimuovi l'opzione *--force*. Se rimuovi l'opzione e il componente aggiuntivo di Amazon EKS deve sovrascrivere la configurazione di un componente aggiuntivo autogestito esistente, la

creazione del componente aggiuntivo di Amazon EKS non riesce e viene visualizzato un messaggio di errore per semplificare la soluzione del conflitto. Prima di specificare questa opzione, assicurati che il componente aggiuntivo di Amazon EKS non gestisca le impostazioni che devi gestire tu, perché questa opzione le sovrascrive.

```
eksctl create addon --cluster my-cluster --name name-of-addon --version latest
 \
   --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
force
```

Puoi vedere un elenco completo di tutte opzioni disponibili per il comando.

```
eksctl create addon --help
```

Per ulteriori informazioni sulle opzioni disponibili, consulta Componenti aggiuntivi nella documentazione di `eksctl`.

AWS Management Console

Per creare un componente aggiuntivo Amazon EKS utilizzando AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui creare il componente aggiuntivo.
3. Seleziona la scheda Componenti aggiuntivi.
4. Scegli Ottieni altri componenti aggiuntivi.
5. Scegli i componenti aggiuntivi da aggiungere al tuo cluster. Puoi scegliere tutti i componenti aggiuntivi di Amazon EKS e i componenti aggiuntivi di Marketplace AWS necessari.

   Per Marketplace AWSquanto riguarda i componenti aggiuntivi, il responsabile IAM che stai utilizzando per creare il componente aggiuntivo deve disporre delle autorizzazioni per il componente aggiuntivo da. AWS LicenseManager AWS LicenseManager richiede un ruolo AWSServiceRoleForAWSLicenseManagerRolecollegato al servizio (SLR) che consente alle AWS risorse di gestire le licenze per tuo conto. L'SLR è un requisito una tantum per singolo account e non dovrai creare SLR separati per ciascun componente aggiuntivo o per ciascun cluster. Per ulteriori informazioni sull'assegnazione di autorizzazioni a un principale IAM,

consulta Aggiunta e rimozione di autorizzazioni per identità IAM nella Guida per l'utente di IAM.

Se i componenti aggiuntivi di Marketplace AWS da installare non sono elencati, puoi cercare i componenti aggiuntivi disponibili inserendo il testo nella casella di ricerca. Nelle Opzioni di filtro, puoi eseguire una ricerca anche per categoria, fornitore o modello di prezzi e quindi scegliere i componenti aggiuntivi dai risultati della ricerca. Dopo aver selezionato i componenti aggiuntivi da installare, scegli Successivo.

6. Nella pagina Configura le impostazioni dei componenti aggiuntivi selezionati:

   - Scegli Visualizza le opzioni di abbonamento per aprire il modulo delle opzioni di abbonamento. Consulta le sezioni Dettagli sui prezzi e Note legali, quindi scegli il pulsante Abbonati per continuare.
   - Per Versione, seleziona la versione che desideri installare. È preferibile la versione contrassegnata come latest (più recente), a meno che il singolo componente aggiuntivo che stai creando non consigli una versione differente. Per determinare se un componente aggiuntivo ha una versione consigliata, consulta la documentazione del componente aggiuntivo che stai creando.
   - Se tutti i componenti aggiuntivi selezionati includono Requires subscription (Richiede sottoscrizione) sotto Status (Stato), seleziona Next (Avanti). Non puoi configurare questi componenti aggiuntivi ulteriormente finché non li hai sottoscritti dopo la creazione del cluster. Per i componenti aggiuntivi che non includono Requires subscription (Richiede sottoscrizione) sotto Status (Stato):
     - per Select IAM role (Seleziona ruolo IAM) accetta l'opzione predefinita, a meno che il componente aggiuntivo non richieda autorizzazioni IAM. Se il componente aggiuntivo richiede AWS autorizzazioni, puoi utilizzare il ruolo IAM del nodo (non impostato) o un ruolo esistente che hai creato per l'utilizzo con il componente aggiuntivo. Se non esiste alcun ruolo da selezionare, non disponi ancora di un ruolo. Indipendentemente dall'opzione scelta, consulta la documentazione del componente aggiuntivo che stai creando per creare una policy IAM e associarla a un ruolo. La selezione di un ruolo IAM richiede la disponibilità di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno per il tuo cluster o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
     - Scegli Optional configuration settings (Impostazioni di configurazione facoltative).
       - Se il componente aggiuntivo richiede una configurazione, inseriscila nella casella Configuration values (Valori di configurazione). Per determinare se il componente

aggiuntivo richiede informazioni di configurazione, consulta la documentazione del componente aggiuntivo che stai creando.

- Seleziona una delle opzioni disponibili per Conflict resolution method (Metodo di risoluzione dei conflitti).

- Seleziona Successivo.

7. Nella pagina Rivedi e aggiungi, scegli Crea. Una volta completata l'installazione del componente aggiuntivo, vengono visualizzati i componenti aggiuntivi installati.
8. Se uno dei componenti aggiuntivi installati richiede una sottoscrizione, completa la procedura seguente:

   1. scegli il pulsante Subscribe (Sottoscrivi) nell'angolo inferiore destro del componente aggiuntivo. Verrai indirizzato alla pagina del componente aggiuntivo in Marketplace AWS. Leggi le informazioni sul componente aggiuntivo, ad esempio Product Overview (Panoramica del prodotto) e Pricing information (Informazioni sui prezzi).
   2. Seleziona il pulsante Continue to Subscribe (Continua la sottoscrizione) nella parte superiore destra della pagina del componente aggiuntivo.
   3. Leggi i Terms and Conditions (Termini e condizioni). Se li accetti, scegli Accept Terms (Accetta i termini). L'elaborazione della sottoscrizione può richiedere vari minuti. Durante l'elaborazione della sottoscrizione, il pulsante Return to Amazon EKS Console (Torna alla console Amazon EKS) è disattivato.
   4. Al termine dell'elaborazione dell'abbonamento, il pulsante Return to Amazon EKS Console (Torna alla console Amazon EKS) non è più disattivato. Scegli il pulsante per tornare alla scheda Add-ons (Componenti aggiuntivi) della console Amazon EKS per il tuo cluster.
   5. Per il componente aggiuntivo che hai sottoscritto, scegli Remove and reinstall (Rimuovi e reinstalla), quindi scegli Reinstall add-on (Reinstalla componente aggiuntivo). L'installazione del componente aggiuntivo può richiedere vari minuti. Una volta completata l'installazione, puoi configurare il componente aggiuntivo.

AWS CLI

Prerequisito

Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o. AWS CloudShell Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati

a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.

Per creare un componente aggiuntivo Amazon EKS utilizzando AWS CLI

1. Determina quali componenti aggiuntivi sono disponibili. Puoi vedere tutti i componenti aggiuntivi disponibili, incluso il tipo e l'autore. Puoi anche vedere l'URL dei componenti aggiuntivi disponibili tramite Marketplace AWS. Sostituisci *1.30* con la versione del cluster.

```
aws eks describe-addon-versions --kubernetes-version 1.30 \
    --query 'addons[].{MarketplaceProductUrl: marketplaceInformation.productUrl,
 Name: addonName, Owner: owner Publisher: publisher, Type: type}' --output table
```

Di seguito viene riportato un output di esempio:

```
-------------------------------------------------------------------------------------------
|
 DescribeAddonVersions
       |
+-----------------------------------------------------------------
+------------------------------+------------------+--------------
+---------------------+
|                      MarketplaceProductUrl                      |
 Name              |      Owner       |  Publisher   |        Type         |
+-----------------------------------------------------------------
+------------------------------+------------------+--------------
+---------------------+
|  None                                                           |  aws-ebs-csi-
driver           |  aws             |  eks         |  storage            |
|  None                                                           |  coredns
                 |  aws             |  eks         |  networking         |
|  None                                                           |  kube-proxy
                 |  aws             |  eks         |  networking         |
|  None                                                           |  vpc-cni
                 |  aws             |  eks         |  networking         |
|  None                                                           |  adot
                 |  aws             |  eks         |  observability      |
```

```
|  https://aws.amazon.com/marketplace/pp/prodview-brb73nceicv7u |
 dynatrace_dynatrace-operator |  aws-marketplace |  dynatrace   |  monitoring
      |
|  https://aws.amazon.com/marketplace/pp/prodview-uhc2iwi5xysoc |
 upbound_universal-crossplane |  aws-marketplace |  upbound     |  infra-
management   |
|  https://aws.amazon.com/marketplace/pp/prodview-hd2ydsrgqy4li |
 teleport_teleport            |  aws-marketplace |  teleport    |  policy-
management  |
|  https://aws.amazon.com/marketplace/pp/prodview-vgghgqdsplhvc |
 factorhouse_kpow             |  aws-marketplace |  factorhouse |  monitoring
      |
|  [...]                                                       |  [...]
               |  [...]           |  [...]       |  [...]              |
+-------------------------------------------------------------
+------------------------------+------------------+--------------
+---------------------+
```

Il tuo output potrebbe essere diverso. In questo output di esempio, sono disponibili tre diversi componenti aggiuntivi di tipo `networking` e cinque componenti aggiuntivi con un autore di tipo `eks`. I componenti aggiuntivi con `aws-marketplace` nella colonna `Owner` potrebbero richiedere una sottoscrizione per poter essere installati. Per ulteriori informazioni sul componente aggiuntivo e per sottoscriverlo, puoi visitare l'URL.

2. Puoi vedere quali versioni sono disponibili per ogni componente aggiuntivo. Sostituisci *`1.30`* con la versione del tuo cluster, quindi sostituisci *`vpc-cni`* con il nome del componente aggiuntivo restituito nella fase precedente.

```
aws eks describe-addon-versions --kubernetes-version 1.30 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

Di seguito viene riportato un output di esempio:

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+---------------------+
| Defaultversion  |       Version       |
+-----------------+---------------------+
|  False          |  v1.12.0-eksbuild.1 |
|  True           |  v1.11.4-eksbuild.1 |
|  False          |  v1.10.4-eksbuild.1 |
```

```
|  False          |  v1.9.3-eksbuild.1   |
+-----------------+----------------------+
```

Per impostazione predefinita, la versione con `True` nella colonna `Defaultversion` è la versione con cui viene creato il componente aggiuntivo.

3. (Facoltativo) Trova le opzioni di configurazione del componente aggiuntivo scelto utilizzando il comando seguente:

```
aws eks describe-addon-configuration --addon-name vpc-cni --addon-
version v1.12.0-eksbuild.1
```

```
{
    "addonName": "vpc-cni",
    "addonVersion": "v1.12.0-eksbuild.1",
    "configurationSchema": "{\"$ref\":\"#/definitions/VpcCni\",\"$schema
\":\"http://json-schema.org/draft-06/schema#\",\"definitions\":{\"Cri\":
{\"additionalProperties\":false,\"properties\":{\"hostPath\":{\"$ref\":
\"#/definitions/HostPath\"}},\"title\":\"Cri\",\"type\":\"object\"},\"Env
\":{\"additionalProperties\":false,\"properties\":{\"ADDITIONAL_ENI_TAGS
\":{\"type\":\"string\"},\"AWS_VPC_CNI_NODE_PORT_SUPPORT\":{\"format\":
\"boolean\",\"type\":\"string\"},\"AWS_VPC_ENI_MTU\":{\"format\":\"integer
\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CONFIGURE_RPFILTER\":{\"format
\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_CUSTOM_NETWORK_CFG\":
{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_EXTERNALSNAT
\":{\"format\":\"boolean\",\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOGLEVEL
\":{\"type\":\"string\"},\"AWS_VPC_K8S_CNI_LOG_FILE\":{\"type
\":\"string\"},\"AWS_VPC_K8S_CNI_RANDOMIZESNAT\":{\"type\":
\"string\"},\"AWS_VPC_K8S_CNI_VETHPREFIX\":{\"type\":\"string
\"},\"AWS_VPC_K8S_PLUGIN_LOG_FILE\":{\"type\":\"string\"},
\"AWS_VPC_K8S_PLUGIN_LOG_LEVEL\":{\"type\":\"string\"},\"DISABLE_INTROSPECTION
\":{\"format\":\"boolean\",\"type\":\"string\"},\"DISABLE_METRICS\":{\"format
\":\"boolean\",\"type\":\"string\"},\"DISABLE_NETWORK_RESOURCE_PROVISIONING
\":{\"format\":\"boolean\",\"type\":\"string\"},\"ENABLE_POD_ENI\":{\"format
\":\"boolean\",\"type\":\"string\"},\"ENABLE_PREFIX_DELEGATION\":{\"format
\":\"boolean\",\"type\":\"string\"},\"WARM_ENI_TARGET\":{\"format\":\"integer
\",\"type\":\"string\"},\"WARM_PREFIX_TARGET\":{\"format\":\"integer\",
\"type\":\"string\"}},\"title\":\"Env\",\"type\":\"object\"},\"HostPath\":
{\"additionalProperties\":false,\"properties\":{\"path\":{\"type\":\"string\"}},
\"title\":\"HostPath\",\"type\":\"object\"},\"Limits\":{\"additionalProperties
\":false,\"properties\":{\"cpu\":{\"type\":\"string\"},\"memory\":{\"type
\":\"string\"}},\"title\":\"Limits\",\"type\":\"object\"},\"Resources\":
```

```
{\"additionalProperties\":false,\"properties\":{\"limits\":{\"$ref\":\"#/
definitions/Limits\"},\"requests\":{\"$ref\":\"#/definitions/Limits\"}},
\"title\":\"Resources\",\"type\":\"object\"},\"VpcCni\":{\"additionalProperties
\":false,\"properties\":{\"cri\":{\"$ref\":\"#/definitions/Cri\"},\"env\":
{\"$ref\":\"#/definitions/Env\"},\"resources\":{\"$ref\":\"#/definitions/
Resources\"}},\"title\":\"VpcCni\",\"type\":\"object\"}}}"
}
```

L'output sarà uno schema JSON standard.

Di seguito è riportato un esempio di valori di configurazione validi, in formato JSON, che funzionano con lo schema precedente.

```
{
  "resources": {
    "limits": {
      "cpu": "100m"
    }
  }
}
```

Di seguito è riportato un esempio di valori di configurazione validi, in formato YAML, che funzionano con lo schema precedente.

```
  resources:
    limits:
      cpu: 100m
```

4. Determina se il componente aggiuntivo richiede le autorizzazioni IAM. In tal caso, è necessario (1) determinare se si desidera utilizzare EKS Pod Identities o IAM Roles for Service Accounts (IRSA), (2) determinare l'ARN del ruolo IAM da utilizzare con il componente aggiuntivo e (3) determinare il nome dell'account di servizio Kubernetes utilizzato dal componente aggiuntivo. Puoi trovare queste informazioni nella documentazione o utilizzando l' AWS API, vedi Recuperare informazioni IAM su un componente aggiuntivo.

   - Amazon EKS suggerisce di utilizzare EKS Pod Identities se il componente aggiuntivo lo supporta. Ciò richiede che Pod Identity Agent sia installato nel cluster. Per ulteriori informazioni sull'utilizzo di Pod Identities con componenti aggiuntivi, consulta. Associa un ruolo IAM a un componente aggiuntivo Amazon EKS utilizzando Pod Identity

- Se il componente aggiuntivo o il cluster non sono configurati per EKS Pod Identities, utilizzate IRSA. Conferma che IRSA sia configurato sul tuo cluster.
- Consulta la documentazione dei componenti aggiuntivi di Amazon EKS per determinare se il componente aggiuntivo richiede le autorizzazioni IAM e il nome dell'account di servizio Kubernetes associato.

5. Crea un componente aggiuntivo di Amazon EKS. Copia il comando seguente sul tuo dispositivo. Apportare le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato:

   - Sostituisci *my-cluster* con il nome del cluster.
   - Sostituisci *vpc-cni* con il nome del componente aggiuntivo da creare restituito nell'output del passaggio precedente.
   - Sostituisci *version-number* con il nome del componente aggiuntivo da utilizzare restituito nell'output della fase precedente.
   - Se il componente aggiuntivo non richiede autorizzazioni IAM, eliminalo. *<service-account-configuration>*
   - Se il componente aggiuntivo (1) richiede le autorizzazioni IAM e (2) il cluster utilizza EKS Pod Identities, sostituiscilo *<service-account-configuration>* con la seguente associazione di identità del pod. *<service-account-name>*Sostituiscilo con il nome dell'account di servizio utilizzato dal componente aggiuntivo. Sostituisci *<role-arn>* con l'ARN di un ruolo IAM. Il ruolo deve avere la politica di fiducia richiesta da EKS Pod Identities.
     - ```
       --pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'
       ```
   - Se il componente aggiuntivo (1) richiede le autorizzazioni IAM e (2) il cluster utilizza IRSA, sostituiscilo *<service-account-configuration>* con la seguente configurazione IRSA. *111122223333*Sostituiscilo con l'ID del tuo account e *role-name* con il nome di un ruolo IAM esistente che hai creato. Per istruzioni sulla creazione del ruolo, consulta la documentazione del componente aggiuntivo che stai creando. Per specificare un ruolo dell'account di servizio, devi disporre di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno per il tuo cluster o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
     - ```
       --service-account-role-arn arn:aws:iam::111122223333:role/role-name
       ```

- Questo comando di esempio sovrascrive l'opzione `--configuration-values` di qualunque versione autogestita esistente del componente aggiuntivo, se presente. Sostituiscilo con i valori di configurazione desiderati, come una stringa o un file di input. Se non desideri fornire valori di configurazione, elimina l'opzione `--configuration-values`. Se non desideri AWS CLI sovrascrivere la configurazione di un componente aggiuntivo autogestito esistente, rimuovi l'opzione. *`--resolve-conflicts OVERWRITE`* Se rimuovi l'opzione e il componente aggiuntivo di Amazon EKS deve sovrascrivere la configurazione di un componente aggiuntivo autogestito esistente, la creazione del componente aggiuntivo di Amazon EKS non riesce e viene visualizzato un messaggio di errore per semplificare la soluzione del conflitto. Prima di specificare questa opzione, assicurati che il componente aggiuntivo di Amazon EKS non gestisca le impostazioni che devi gestire tu, perché questa opzione le sovrascrive.

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    <service-account-configuration> --configuration-values '{"resources":
{"limits":{"cpu":"100m"}}}' --resolve-conflicts OVERWRITE
```

```
aws eks create-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    <service-account-configuration> --configuration-values 'file://example.yaml'
 --resolve-conflicts OVERWRITE
```

Per un elenco completo delle opzioni disponibili, consulta `create-addon` nella Documentazione di riferimento della riga di comando Amazon EKS. Se il componente aggiuntivo che hai creato include `aws-marketplace` elencato nella colonna `Owner` di un passaggio precedente, la creazione potrebbe non riuscire e potresti ricevere un messaggio di errore simile al seguente.

```
{
    "addon": {
        "addonName": "addon-name",
        "clusterName": "my-cluster",
        "status": "CREATE_FAILED",
        "addonVersion": "version",
        "health": {
            "issues": [
                {
```

```
                    "code": "AddonSubscriptionNeeded",
                    "message": "You are currently not subscribed to this add-
on. To subscribe, visit the Marketplace AWS console, agree to the seller EULA,
 select the pricing type if required, then re-install the add-on"
[...]
```

Se ricevi un errore simile a quello nell'output precedente, visita l'URL nell'output di un passaggio precedente per sottoscrivere il componente aggiuntivo. Dopo la sottoscrizione, esegui nuovamente il comando `create-addon`.

Aggiornamento di un componente aggiuntivo

Amazon EKS non aggiorna automaticamente il componente aggiuntivo quando vengono rilasciate nuove versioni o dopo l'aggiornamento del cluster a una nuova versione secondaria di Kubernetes. Per aggiornare un componente aggiuntivo per un cluster esistente, devi avviare l'aggiornamento. Dopo l'avvio dell'aggiornamento, Amazon EKS aggiorna automaticamente il componente aggiuntivo. Prima di aggiornare un componente aggiuntivo, consulta la documentazione attuale del componente aggiuntivo. Per un elenco dei componenti aggiuntivi disponibili, consulta Componenti aggiuntivi Amazon EKS disponibili da Amazon EKS. Se il componente aggiuntivo richiede un ruolo IAM, consulta i dettagli del componente aggiuntivo specifico alla pagina Componenti aggiuntivi Amazon EKS disponibili da Amazon EKS per i dettagli sulla creazione del ruolo.

Puoi aggiornare un componente aggiuntivo Amazon EKS utilizzando`eksctl`, il AWS Management Console, o il AWS CLI.

eksctl

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Come aggiornare un componente aggiuntivo di Amazon EKS utilizzando **`eksctl`**

1. Determina i componenti aggiuntivi e le versioni dei componenti aggiuntivi installati sul tuo cluster. Sostituisci *`my-cluster`* con il nome del cluster.

```
eksctl get addon --cluster my-cluster
```

Di seguito viene riportato un output di esempio:

```
NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
coredns     v1.8.7-eksbuild.2    ACTIVE  0
kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0                v1.23.8-eksbuild.2
vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0                v1.12.0-
eksbuild.1,v1.11.4-eksbuild.1,v1.11.3-eksbuild.1,v1.11.2-eksbuild.1,v1.11.0-
eksbuild.1
```

L'output potrebbe avere un aspetto diverso in base ai componenti aggiuntivi e alle versioni presenti nel tuo cluster. Nell'output di esempio precedente, due componenti aggiuntivi esistenti nel cluster hanno versioni più recenti disponibili nella colonna `UPDATE AVAILABLE`.

2. Aggiorna il componente aggiuntivo.

   1. Copia il comando seguente sul tuo dispositivo. Apporta le seguenti modifiche al comando in base alla necessità:

      - Sostituisci *`my-cluster`* con il nome del cluster.
      - *`region-code`*Sostituiscilo con Regione AWS quello in cui si trova il cluster.
      - Sostituisci *`vpc-cni`* con il nome del componente aggiuntivo da aggiornare restituito nell'output del passaggio precedente.
      - Se desideri aggiornare una versione del componente aggiuntivo precedente all'ultima versione, sostituisci *`latest`* con il numero di versione che vuoi utilizzare e che è stato restituito nell'output di un passaggio precedente. Alcuni componenti aggiuntivi includono versioni consigliate. Per ulteriori informazioni, consulta la documentazione del componente aggiuntivo che stai aggiornando.
      - Se il componente aggiuntivo utilizza un account di servizio Kubernetes e un ruolo IAM, sostituisci *`111122223333`* con l'ID del tuo account e *`role-name`* con il nome di un ruolo IAM esistente che hai creato. Per istruzioni sulla creazione del ruolo, consulta la documentazione del componente aggiuntivo che stai creando. Per specificare un ruolo dell'account di servizio, devi disporre di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno per il tuo cluster o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.

        Se il componente aggiuntivo non utilizza un account di servizio Kubernetes e un ruolo IAM, elimina la riga **`serviceAccountRoleARN: arn:aws:iam::111122223333:role/role-name`**.

- L'*preserve*opzione conserva i valori esistenti per il componente aggiuntivo. Se hai impostato valori personalizzati per le impostazioni del componente aggiuntivo e non utilizzi questa opzione, Amazon EKS sovrascrive i tuoi valori con quelli predefiniti. Se utilizzi questa opzione, è preferibile testare eventuali modifiche ai campi e ai valori su un cluster non di produzione prima di aggiornare il componente aggiuntivo sul cluster di produzione. Se modifichi questo valore in `overwrite`, tutte le impostazioni vengono modificate nei valori predefiniti di Amazon EKS. Se hai impostato valori personalizzati per un'impostazione qualunque, è possibile che vengano sovrascritti con i valori predefiniti di Amazon EKS. Se modifichi questo valore in `none`, Amazon EKS non modifica il valore di alcuna impostazione, ma l'aggiornamento potrebbe non riuscire. Se l'aggiornamento non riesce, riceverai un messaggio di errore che ti aiuterà a risolvere il conflitto.

```
cat >update-addon.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig
metadata:
  name: my-cluster
  region: region-code

addons:
- name: vpc-cni
  version: latest
  serviceAccountRoleARN: arn:aws:iam::111122223333:role/role-name
  resolveConflicts: preserve
EOF
```

2. Esegui il comando modificato per creare il file `update-addon.yaml`.
3. Applica il file di configurazione al tuo cluster.

```
eksctl update addon -f update-addon.yaml
```

Per ulteriori informazioni sulle opzioni disponibili, consulta Addons (Componenti aggiuntivi) nella documentazione di `eksctl`.

AWS Management Console

Per aggiornare un componente aggiuntivo Amazon EKS utilizzando AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui configurare il componente aggiuntivo.
3. Seleziona la scheda Componenti aggiuntivi.
4. Seleziona la casella nella parte superiore destra della casella del componente aggiuntivo e scegli Edit (Modifica).
5. Nella pagina Configure ***name of addon*** (Configura il nome del componente aggiuntivo):
   - seleziona la Version (Versione) da utilizzare. Il componente aggiuntivo potrebbe avere una versione raccomandata. Per ulteriori informazioni, consulta la documentazione del componente aggiuntivo che stai aggiornando.
   - Per il ruolo Select IAM, puoi utilizzare il ruolo IAM del nodo (non impostato) o un ruolo esistente che hai creato per l'utilizzo con il componente aggiuntivo. Se non esiste alcun ruolo da selezionare, non disponi ancora di un ruolo. Indipendentemente dall'opzione scelta, consulta la documentazione del componente aggiuntivo che stai creando per creare una policy IAM e associarla a un ruolo. La selezione di un ruolo IAM richiede la disponibilità di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno per il tuo cluster o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.
   - Su `Code editor` inserisci qualsiasi informazione di configurazione specifica del componente aggiuntivo. Per ulteriori informazioni, consulta la documentazione del componente aggiuntivo che stai aggiornando.
   - Per Metodo di risoluzione dei conflitti, seleziona una delle opzioni disponibili. Se hai impostato valori personalizzati per le impostazioni del componente aggiuntivo, è preferibile l'opzione Preserve (Conserva). Se non scegli questa opzione, Amazon EKS sovrascrive i tuoi valori con quelli predefiniti. Se utilizzi questa opzione, è preferibile testare eventuali modifiche ai campi e ai valori su un cluster non di produzione prima di aggiornare il componente aggiuntivo sul cluster di produzione.
6. Scegli Aggiorna.

AWS CLI

Prerequisito

Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o AWS CloudShell. Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.

Per aggiornare un componente aggiuntivo Amazon EKS utilizzando AWS CLI

1. Visualizza l'elenco dei componenti aggiuntivi installati. Sostituisci *`my-cluster`* con il nome del cluster.

   ```
   aws eks list-addons --cluster-name my-cluster
   ```

   Di seguito viene riportato un output di esempio:

   ```
   {
       "addons": [
           "coredns",
           "kube-proxy",
           "vpc-cni"
       ]
   }
   ```

2. Visualizza la versione corrente del componente aggiuntivo da aggiornare. Sostituisci *`my-cluster`* con il nome del tuo cluster e *`vpc-cni`* con il nome del componente aggiuntivo da aggiornare.

   ```
   aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni --query
    "addon.addonVersion" --output text
   ```

   Di seguito viene riportato un output di esempio:

```
v1.10.4-eksbuild.1
```

3. Puoi visualizzare le versioni disponibili del componente aggiuntivo per la versione del tuo cluster. Sostituisci *1.30* con il nome del tuo cluster e *vpc-cni* con il nome del componente aggiuntivo da aggiornare.

```
aws eks describe-addon-versions --kubernetes-version 1.30 --addon-name vpc-cni \
    --query 'addons[].addonVersions[].{Version: addonVersion, Defaultversion:
 compatibilities[0].defaultVersion}' --output table
```

Di seguito viene riportato un output di esempio:

```
-----------------------------------------
|          DescribeAddonVersions         |
+-----------------+----------------------+
| Defaultversion  |       Version        |
+-----------------+----------------------+
|  False          |  v1.12.0-eksbuild.1  |
|  True           |  v1.11.4-eksbuild.1  |
|  False          |  v1.10.4-eksbuild.1  |
|  False          |  v1.9.3-eksbuild.1   |
+-----------------+----------------------+
```

Per impostazione predefinita, la versione con `True` nella colonna `Defaultversion` è la versione con cui viene creato il componente aggiuntivo.

4. Aggiorna il componente aggiuntivo. Copia il comando seguente sul tuo dispositivo. Apporta le seguenti modifiche al comando, se necessario, quindi esegui il comando modificato.

- Sostituisci *my-cluster* con il nome del cluster.
- Sostituisci *vpc-cni* con il nome del componente aggiuntivo da aggiornare che è stato restituito nell'output di un passaggio precedente.
- Sostituisci *version-number* con la versione a cui eseguire l'aggiornamento restituita nell'output del passaggio precedente. Alcuni componenti aggiuntivi includono versioni consigliate. Per ulteriori informazioni, consulta la documentazione del componente aggiuntivo che stai aggiornando.
- Se il componente aggiuntivo utilizza un account di servizio Kubernetes e un ruolo IAM, sostituisci *111122223333* con l'ID del tuo account e *role-name* con il nome di un

ruolo IAM esistente che hai creato. Per istruzioni sulla creazione del ruolo, consulta la documentazione del componente aggiuntivo che stai creando. Per specificare un ruolo dell'account di servizio, devi disporre di un provider IAM OpenID Connect (OIDC) per il tuo cluster. Per stabilire se ne possiedi uno per il tuo cluster o per crearne uno, consulta Crea un OIDC provider IAM per il tuo cluster.

Se il componente aggiuntivo non utilizza un account di servizio Kubernetes e un ruolo IAM, elimina la riga **`serviceAccountRoleARN: arn:aws:iam::`*`111122223333`*`:role/`*`role-name`***.

- L'opzione **`--resolve-conflicts`** *`PRESERVE`* (CONSERVA) mantiene i valori esistenti per il componente aggiuntivo. Se hai impostato valori personalizzati per le impostazioni del componente aggiuntivo e non utilizzi questa opzione, Amazon EKS sovrascrive i tuoi valori con quelli predefiniti. Se utilizzi questa opzione, è preferibile testare eventuali modifiche ai campi e ai valori su un cluster non di produzione prima di aggiornare il componente aggiuntivo sul cluster di produzione. Se modifichi questo valore in `overwrite`, tutte le impostazioni vengono modificate nei valori predefiniti di Amazon EKS. Se hai impostato valori personalizzati per un'impostazione qualunque, è possibile che vengano sovrascritti con i valori predefiniti di Amazon EKS. Se modifichi questo valore in `none`, Amazon EKS non modifica il valore di alcuna impostazione, ma l'aggiornamento potrebbe non riuscire. Se l'aggiornamento non riesce, riceverai un messaggio di errore che ti aiuterà a risolvere il conflitto.

- Se desideri rimuovere tutte le configurazioni personalizzate, esegui l'aggiornamento utilizzando l'opzione *`--configuration-values '{}'`*. In tal modo, verranno ripristinati i valori predefiniti di tutte le configurazioni personalizzate. Se non desideri modificare la configurazione personalizzata, non fornire il flag *`--configuration-values`*. Per modificare una configurazione personalizzata, sostituisci *`{}`* con i nuovi parametri. Per visualizzare un elenco di parametri, consulta la fase Visualizzazione dello schema di configurazione nella sezione Creazione di un componente aggiuntivo.

```
aws eks update-addon --cluster-name my-cluster --addon-name vpc-cni --addon-
version version-number \
    --service-account-role-arn arn:aws:iam::111122223333:role/role-name --
configuration-values '{}' --resolve-conflicts PRESERVE
```

5. Controlla lo stato dell'aggiornamento. Sostituisci *my-cluster* con il nome del tuo cluster e *vpc-cni* con il nome del componente aggiuntivo da aggiornare.

```
aws eks describe-addon --cluster-name my-cluster --addon-name vpc-cni
```

Di seguito viene riportato un output di esempio:

```
{
    "addon": {
        "addonName": "vpc-cni",
        "clusterName": "my-cluster",
        "status": "UPDATING",
[...]
```

L'aggiornamento è completo quando lo stato è `ACTIVE`.

Aggiornamento di un componente aggiuntivo

Quando elimini un componente aggiuntivo di Amazon EKS:

- non sono previste interruzioni per la funzionalità fornita dal componente aggiuntivo.
- Se utilizzi IAM Roles for Service Accounts (IRSA) e al componente aggiuntivo è associato un ruolo IAM, il ruolo IAM non viene rimosso.
- Se utilizzi Ipod Identities, tutte le Associazioni di identità Pod di proprietà del componente aggiuntivo vengono eliminate. Se si specifica l'`--preserve`opzione su, le associazioni vengono mantenute AWS CLI.
- Amazon EKS interrompe la gestione delle impostazioni per il componente aggiuntivo.
- La console smette di segnalarti la disponibilità di nuove versioni.
- Non puoi aggiornare il componente aggiuntivo utilizzando AWS strumenti o API.
- Puoi scegliere di lasciare il software del componente aggiuntivo nel tuo cluster in modo da poterlo gestire autonomamente, oppure puoi rimuoverlo. È opportuno rimuovere il software del componente aggiuntivo dal tuo cluster se non esistono risorse sul cluster che dipendono dalla funzionalità fornita dal componente aggiuntivo.

Puoi eliminare un componente aggiuntivo di Amazon EKS dal cluster utilizzando `eksctl`, la AWS Management Console o la AWS CLI.

eksctl

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Come eliminare un componente aggiuntivo di Amazon EKS utilizzando **`eksctl`**

1. Determina i componenti aggiuntivi installati sul tuo cluster. Sostituisci *`my-cluster`* con il nome del cluster.

   ```
   eksctl get addon --cluster my-cluster
   ```

   Di seguito viene riportato un output di esempio:

   ```
   NAME        VERSION              STATUS  ISSUES  IAMROLE  UPDATE AVAILABLE
   coredns     v1.8.7-eksbuild.2    ACTIVE  0
   kube-proxy  v1.23.7-eksbuild.1   ACTIVE  0
   vpc-cni     v1.10.4-eksbuild.1   ACTIVE  0
   [...]
   ```

   L'output potrebbe variare a seconda dei componenti aggiuntivi e delle versioni presenti sul tuo cluster.

2. Elimina il componente aggiuntivo. Sostituisci *`my-cluster`* con il nome del cluster e *`name-of-add-on`* con il nome del componente aggiuntivo da rimuovere restituito nell'output del passaggio precedente. Se rimuovi l'opzione ***`--preserve`***, oltre al fatto che Amazon EKS non gestisce più il componente aggiuntivo, il software del componente aggiuntivo viene rimosso dal tuo cluster.

   ```
   eksctl delete addon --cluster my-cluster --name name-of-addon --preserve
   ```

AWS Management Console

Per eliminare un componente aggiuntivo Amazon EKS utilizzando il AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.

2. Nel riquadro di navigazione a sinistra, seleziona Cluster, quindi seleziona il nome del cluster per cui rimuovere il componente aggiuntivo di Amazon EKS.
3. Seleziona la scheda Componenti aggiuntivi.
4. Seleziona la casella di controllo nella parte superiore destra della casella del componente aggiuntivo e scegli Remove (Rimuovi). Seleziona Preserve on the cluster (Conserva sul cluster) se desideri che Amazon EKS interrompa la gestione delle impostazioni per il componente aggiuntivo, ma desideri mantenere il software del componente aggiuntivo sul tuo cluster in modo da poter gestire autonomamente tutte le impostazioni del componente aggiuntivo. Digita il nome del componente aggiuntivo e seleziona Remove (Rimuovi).

AWS CLI

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Per eliminare un componente aggiuntivo Amazon EKS utilizzando il AWS CLI

1. Visualizza l'elenco dei componenti aggiuntivi installati. Sostituisci *my-cluster* con il nome del cluster.

```
aws eks list-addons --cluster-name my-cluster
```

Di seguito viene riportato un output di esempio:

```
{
    "addons": [
        "coredns",
        "kube-proxy",
        "vpc-cni",
        "name-of-addon"
    ]
}
```

2. Elimina il componente aggiuntivo installato. Sostituisci *my-cluster* con il nome del tuo cluster e *name-of-add-on* con il nome del componente aggiuntivo da rimuovere. La rimozione di ***--preserve*** rimuove il software del componente aggiuntivo dal cluster.

```
aws eks delete-addon --cluster-name my-cluster --addon-name name-of-addon --
preserve
```

Di seguito è riportato un output di esempio abbreviato.

```
{
    "addon": {
        "addonName": "name-of-add-on",
        "clusterName": "my-cluster",
        "status": "DELETING",
[...]
```

3. Controlla lo stato dell'eliminazione. Sostituisci *my-cluster* con il nome del tuo cluster e *name-of-addon* con il nome del componente aggiuntivo da rimuovere.

```
aws eks describe-addon --cluster-name my-cluster --addon-name name-of-addon
```

Di seguito è riportato l'output di esempio dopo l'eliminazione del componente.

```
An error occurred (ResourceNotFoundException) when calling the DescribeAddon
 operation: No addon: name-of-addon found in cluster: my-cluster
```

Recupera la compatibilità delle versioni del componente aggiuntivo

Utilizza l'describe-addon-verisionsAPI per elencare le versioni disponibili dei componenti aggiuntivi EKS e le versioni di Kubernetes supportate da ciascuna versione del componente aggiuntivo.

Recupera la compatibilità delle versioni del componente aggiuntivo ()AWS CLI

1. Verifica che AWS CLI sia installato e funzionante con. `aws sts get-caller-identity` Se questo comando non funziona, scopri come iniziare con AWS CLI.
2. Determina il nome del componente aggiuntivo per il quale desideri recuperare le informazioni sulla compatibilità delle versioni, ad esempio. `amazon-cloudwatch-observability`
3. Determina la versione Kubernetes del tuo cluster, ad esempio. `1.28`
4. Usa il AWS CLI per recuperare le versioni del componente aggiuntivo compatibili con la versione Kubernetes del tuo cluster.

```
aws eks describe-addon-versions --addon-name amazon-cloudwatch-observability --
kubernetes-version 1.29
```

Di seguito viene riportato un output di esempio:

```
{
    "addons": [
        {
            "addonName": "amazon-cloudwatch-observability",
            "type": "observability",
            "addonVersions": [
                {
                    "addonVersion": "v1.5.0-eksbuild.1",
                    "architecture": [
                        "amd64",
                        "arm64"
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.28",
                            "platformVersions": [
                                "*"
                            ],
                            "defaultVersion": true
                        }
                    ],
[...]
```

Questo output mostra che la versione del componente aggiuntivo `v1.5.0-eksbuild.1` è compatibile con la versione del cluster Kubernetes. `1.28`

Gestione dei campi Kubernetes

I componenti aggiuntivi Amazon EKS vengono installati nel cluster utilizzando configurazioni standard e best practice. Per ulteriori informazioni sull'aggiunta di un componente aggiuntivo di Amazon EKS al cluster, consulta Componenti aggiuntivi Amazon EKS.

Potresti voler personalizzare la configurazione di un componente aggiuntivo Amazon EKS per abilitare funzionalità avanzate. Amazon EKS utilizza la funzionalità di applicazione lato server

Kubernetes per abilitare la gestione di un componente aggiuntivo da parte di Amazon EKS senza sovrascrivere la configurazione per le impostazioni non gestite da Amazon EKS. Per ulteriori informazioni, consulta Applicazione lato server nella documentazione di Kubernetes. Per raggiungere questo obiettivo, Amazon EKS gestisce un insieme minimo di campi per ogni componente aggiuntivo che installa. Puoi modificare senza problemi tutti i campi non gestiti da Amazon EKS o da un altro processo del piano di controllo Kubernetes, come `kube-controller-manager`.

> **Important**
>
> La modifica di un campo gestito da Amazon EKS impedisce ad Amazon EKS di gestire il componente aggiuntivo e può comportare la sovrascrittura delle modifiche quando un componente aggiuntivo viene aggiornato.

Visualizzazione dello stato della gestione dei campi

È possibile utilizzare `kubectl` per vedere quali campi sono gestiti da Amazon EKS per qualsiasi componente aggiuntivo Amazon EKS.

Per visualizzare lo stato di gestione di un campo

1. Determinare quale componente aggiuntivo si desidera esaminare. Per visualizzare tutte le `deployments` e i `DaemonSets` implementati nel cluster, consulta Visualizzazione delle risorse Kubernetes.
2. Visualizzazione dei campi gestiti per un componente aggiuntivo eseguendo il comando seguente:

   ```
   kubectl get type/add-on-name -n add-on-namespace -o yaml
   ```

   Ad esempio, è possibile visualizzare i campi gestiti per il componente aggiuntivo CoreDNS con il comando seguente.

   ```
   kubectl get deployment/coredns -n kube-system -o yaml
   ```

   La gestione dei campi è elencata nella sezione seguente dell'output restituito.

   ```
   [...]
   managedFields:
     - apiVersion: apps/v1
       fieldsType: FieldsV1
   ```

```
    fieldsV1:
[...]
```

> **Note**
>
> Se non visualizzi `managedFields` nell'output, aggiungi **`--show-managed-fields`** al comando ed eseguilo di nuovo. La versione di `kubectl` che stai utilizzando determina se i campi gestiti vengono restituiti per impostazione predefinita.

Informazioni sulla sintassi di gestione dei campi nell'API Kubernetes

Quando visualizzi i dettagli di un oggetto Kubernetes, nell'output vengono restituiti sia i campi gestiti che quelli non gestiti. I campi gestiti possono essere dei seguenti tipi:

- Completamente gestito – Tutte le chiavi del campo sono gestite da Amazon EKS. Le modifiche a qualsiasi valore causano un conflitto.
- Parzialmente gestiti – Alcune chiavi per il campo sono gestite da Amazon EKS. Solo le modifiche alle chiavi gestite esplicitamente da Amazon EKS causano un conflitto.

Entrambi i tipi di campi sono taggati con `manager: eks`.

Ogni chiave è un `.` che rappresenta il campo stesso, che viene sempre mappato a un insieme vuoto o a una stringa che rappresenta un sottocampo o un elemento. L'output per la gestione del campo è costituito dai seguenti tipi di dichiarazioni:

- `f:`*`name`*, dove *`name`* è il nome di un campo in un elenco.
- `k:`*`keys`*, dove *`keys`* è una mappa dei campi di una voce di elenco.
- `v:`*`value`*, dove *`value`* è il valore formattato JSON esatto di una elemento in elenco.
- `i:`*`index`*, dove *`index`* è la posizione di un elemento nell'elenco.

Le seguenti porzioni di output per il componente aggiuntivo CoreDNS illustrano le dichiarazioni precedenti:

- Campi completamente gestiti – Se un campo gestito ha un `f:` (campo) specificato, ma non `k:` (chiave), l'intero campo è gestito. Le modifiche apportate a qualsiasi valore in questo campo causano un conflitto.

Nel seguente output, è possibile vedere che il container denominato `coredns` è gestito da `eks`. La `args`, `image`, e i sottocampi `imagePullPolicy` sono gestiti anche da `eks`. Le modifiche a qualsiasi valore in questi campi causano un conflitto.

```
[...]
f:containers:
  k:{"name":"coredns"}:
  .: {}
  f:args: {}
  f:image: {}
  f:imagePullPolicy: {}
[...]
manager: eks
[...]
```

- Campi parzialmente gestiti – Se una chiave gestita ha un valore specificato, le chiavi dichiarate vengono gestite per quel campo. La modifica delle chiavi specificate causa un conflitto.

  Nel seguente output, è possibile vedere che `eks` gestisce il `config-volume` e volumi `tmp` impostati con la chiave `name`.

```
[...]
f:volumes:
  k:{"name":"config-volume"}:
    .: {}
    f:configMap:
      f:items: {}
      f:name: {}
    f:name: {}
  k:{"name":"tmp"}:
    .: {}
    f:name: {}
[...]
manager: eks
[...]
```

- Aggiunta di chiavi a campi parzialmente gestiti – Se viene gestita solo una chiave-valore specifica, è possibile aggiungere ulteriori chiavi, ad esempio, argomenti, a un campo senza causare conflitti. Se aggiungi altre chiavi, assicurati per prima cosa che il campo non sia gestito. L'aggiunta o la modifica di qualsiasi valore gestito causa un conflitto.

Nel seguente output, è possibile vedere come sia la chiave `name` che il campo `name` siano gestiti. L'aggiunta o la modifica di qualsiasi nome container causa un conflitto con questa chiave gestita.

```
[...]
f:containers:
  k:{"name":"coredns"}:
[...]
    f:name: {}
[...]
manager: eks
[...]
```

Associa un ruolo IAM a un componente aggiuntivo Amazon EKS utilizzando Pod Identity

Alcuni componenti aggiuntivi di Amazon EKS richiedono le autorizzazioni dei ruoli IAM per chiamare AWS le API. Ad esempio, il componente aggiuntivo Amazon VPC CNI richiama determinate AWS API per configurare le risorse di rete nel tuo account. A questi componenti aggiuntivi deve essere concessa l'autorizzazione tramite IAM. AWS Più specificamente, l'account di servizio del pod su cui è in esecuzione il componente aggiuntivo deve essere associato a un ruolo IAM con una policy IAM sufficiente.

Il modo consigliato per concedere AWS le autorizzazioni ai carichi di lavoro del cluster è utilizzare la funzionalità Pod Identities di Amazon EKS. Puoi utilizzare una Pod Identity Association per mappare l'account di servizio di un componente aggiuntivo a un ruolo IAM. Se un pod utilizza un account di servizio con un'associazione, Amazon EKS imposta le variabili di ambiente nei container del pod. Le variabili di ambiente configurano gli AWS SDK, inclusa la AWS CLI, per utilizzare le credenziali EKS Pod Identity. Scopri di più su EKS Pod Identities.

I componenti aggiuntivi Amazon EKS possono aiutare a gestire il ciclo di vita delle associazioni di identità dei pod corrispondenti al componente aggiuntivo. Ad esempio, puoi creare o aggiornare un componente aggiuntivo Amazon EKS e l'associazione di identità del pod necessaria in una singola chiamata API. Amazon EKS fornisce anche un'API per il recupero delle policy IAM suggerite.

Utilizzo consigliato:

1. Verifica che l'agente di identità del pod Amazon EKS sia configurato sul tuo cluster.

2. Stabilisci se il componente aggiuntivo che desideri installare richiede le autorizzazioni IAM utilizzando l'`describe-addon-versions` AWS CLI operazione. Se il `requiresIamPermissions` flag è`true`, allora dovresti usare l'`describe-addon-configuration`operazione per determinare le autorizzazioni necessarie all'addon. La risposta include un elenco di politiche IAM gestite suggerite.
3. Recupera il nome dell'account di servizio Kubernetes e la policy IAM suggerita utilizzando l'operazione CLI. `describe-addon-configuration` Valuta l'ambito della policy suggerita rispetto ai tuoi requisiti di sicurezza.
4. Crea un ruolo IAM utilizzando la politica di autorizzazioni suggerita e la politica di fiducia richiesta da Pod Identity. Per ulteriori informazioni, consulta Creazione dell'associazione EKS Pod Identity.
5. Crea o aggiorna un componente aggiuntivo Amazon EKS utilizzando la CLI. Specificare almeno un'associazione di identità del pod. Un'associazione di identità pod è (1) il nome di un account di servizio Kubernetes e (2) l'ARN di un ruolo IAM.

Considerazioni:

- Le associazioni di identità dei pod create utilizzando le API del componente aggiuntivo sono di proprietà del rispettivo componente aggiuntivo. Se si elimina il componente aggiuntivo, viene eliminata anche l'associazione di identità del pod. Puoi impedire questa eliminazione a cascata utilizzando l'`preserve`opzione durante l'eliminazione di un componente aggiuntivo utilizzando l'API or. AWS CLI Se necessario, puoi anche aggiornare o eliminare direttamente l'associazione di identità del pod. I componenti aggiuntivi non possono assumere la proprietà delle associazioni di identità dei pod esistenti. È necessario eliminare l'associazione esistente e ricrearla utilizzando un'operazione di creazione o aggiornamento del componente aggiuntivo.
- Amazon EKS consiglia di utilizzare le associazioni di identità dei pod per gestire le autorizzazioni IAM per i componenti aggiuntivi. Il metodo precedente, IAM roles for service accounts (IRSA), è ancora supportato. È possibile specificare sia un'associazione IRSA `serviceAccountRoleArn` che un'associazione di identità del pod per un componente aggiuntivo. Se l'agente di identità del pod EKS è installato nel cluster, `serviceAccountRoleArn` verrà ignorato e EKS utilizzerà l'associazione di identità del pod fornita. Se Pod Identity non è abilitato, `serviceAccountRoleArn` verrà utilizzato.
- Se aggiorni le associazioni di identità dei pod per un componente aggiuntivo esistente, Amazon EKS avvia un riavvio progressivo dei pod aggiuntivi.

Recupera informazioni IAM su un componente aggiuntivo

Puoi utilizzare il AWS CLI per determinare (1) se un componente aggiuntivo richiede le autorizzazioni IAM e (2) una politica IAM suggerita per quel componente aggiuntivo.

Recupera informazioni IAM su un componente aggiuntivo Amazon EKS ()AWS CLI

1. Determina il nome del componente aggiuntivo che desideri installare e la versione Kubernetes del cluster. Scopri di più sui componenti aggiuntivi Amazon EKS disponibili.
2. Utilizza il AWS CLI per determinare se il componente aggiuntivo richiede le autorizzazioni IAM.

```
aws eks describe-addon-versions \
--addon-name <addon-name> \
--kubernetes-version <kubernetes-version>
```

Per esempio:

```
aws eks describe-addon-versions \
--addon-name aws-ebs-csi-driver \
--kubernetes-version 1.30
```

Esamina il seguente output di esempio. Nota che `requiresIamPermissions` è `true` la versione del componente aggiuntivo predefinita. È necessario specificare la versione del componente aggiuntivo quando si recupera la policy IAM consigliata.

```
{
    "addons": [
        {
            "addonName": "aws-ebs-csi-driver",
            "type": "storage",
            "addonVersions": [
                {
                    "addonVersion": "v1.31.0-eksbuild.1",
                    "architecture": [
                        "amd64",
                        "arm64"
                    ],
                    "compatibilities": [
                        {
                            "clusterVersion": "1.30",
                            "platformVersions": [
```

```
                                "*"
                            ],
                            "defaultVersion": true
                        }
                    ],
                    "requiresConfiguration": false,
                    "requiresIamPermissions": true
                },
[...]
```

3. Se il componente aggiuntivo richiede le autorizzazioni IAM, utilizza AWS CLI per recuperare una policy IAM consigliata.

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name <addon-name> \
--addon-version <addon-version>
```

Per esempio:

```
aws eks describe-addon-configuration \
--query podIdentityConfiguration \
--addon-name aws-ebs-csi-driver \
--addon-version v1.31.0-eksbuild.1
```

Esamina il seguente risultato. Prendi nota di `recommendedManagedPolicies`.

```
[
    {
        "serviceAccount": "ebs-csi-controller-sa",
        "recommendedManagedPolicies": [
            "arn:aws:iam::aws:policy/service-role/AmazonEBSCSIDriverPolicy"
        ]
    }
]
```

4. Crea un ruolo IAM e allega la Managed Policy consigliata. In alternativa, esamina la policy gestita e definisci le autorizzazioni in modo appropriato. Consulta le istruzioni per creare un ruolo IAM da utilizzare con EKS Pod Identities.

Aggiorna il componente aggiuntivo con IAM Role

Aggiorna un componente aggiuntivo Amazon EKS per utilizzare una Pod Identity Association ()AWS CLI

1. Determina:
   - `cluster-name`— Il nome del cluster EKS su cui installare il componente aggiuntivo.
   - `addon-name`— Il nome del componente aggiuntivo Amazon EKS da installare.
   - `service-account-name`— Il nome dell'account di servizio Kubernetes utilizzato dal componente aggiuntivo.
   - `iam-role-arn`— L'ARN di un ruolo IAM con autorizzazioni sufficienti per il componente aggiuntivo. Il ruolo IAM deve avere la politica di fiducia richiesta per EKS Pod Identity.
2. Aggiorna il componente aggiuntivo utilizzando la AWS CLI. Puoi anche specificare le associazioni di identità dei pod durante la creazione di un componente aggiuntivo, utilizzando la stessa sintassi. `--pod-identity-assocations` Tieni presente che quando specifichi le associazioni di identità dei pod durante l'aggiornamento di un componente aggiuntivo, tutte le associazioni di identità dei pod precedenti vengono sovrascritte.

   ```
   aws eks update-addon --cluster-name <cluster-name> \
   --addon-name <addon-name> \
   --pod-identity-associations 'serviceAccount=<service-account-name>,roleArn=<role-arn>'
   ```

   Per esempio:

   ```
   aws eks update-addon --cluster-name mycluster \
   --addon-name aws-ebs-csi-driver \
   --pod-identity-associations 'serviceAccount=ebs-csi-controller-sa,roleArn=arn:aws:iam::123456789012:role/StorageDriver'
   ```
3. Convalida che l'associazione di identità del pod sia stata creata:

   ```
   aws eks list-pod-identity-associations --cluster-name <cluster-name>
   ```

   A operazione riuscita, viene visualizzato un output simile al seguente. Nota il ownerARN del componente aggiuntivo EKS.

```
{
    "associations": [
        {
            "clusterName": "mycluster",
            "namespace": "kube-system",
            "serviceAccount": "ebs-csi-controller-sa",
            "associationArn": "arn:aws:eks:us-
west-2:123456789012:podidentityassociation/mycluster/a-4wvljrezsukshq1bv",
            "associationId": "a-4wvljrezsukshq1bv",
            "ownerArn": "arn:aws:eks:us-west-2:123456789012:addon/mycluster/aws-
ebs-csi-driver/9cc7ce8c-2e15-b0a7-f311-426691cd8546"
        }
    ]
}
```

Rimuovi le associazioni dal componente aggiuntivo

Rimuovi tutte le associazioni di identità dei pod da un componente aggiuntivo Amazon EKS ()AWS CLI

1. Determina:
   - `cluster-name`— Il nome del cluster EKS su cui installare il componente aggiuntivo.
   - `addon-name`— Il nome del componente aggiuntivo Amazon EKS da installare.
2. Aggiorna l'addon per specificare un array vuoto di associazioni di identità dei pod.

```
aws eks update-addon --cluster-name <cluster-name> \
--addon-name <addon-name> \
--pod-identity-associations "[]"
```

Risolvi i problemi relativi a Pod Identities for EKS Add-ons

Se i componenti aggiuntivi riscontrano errori durante il tentativo di eseguire operazioni su AWS API, SDK o CLI, conferma quanto segue:

- Il Pod Identity Agent è installato nel tuo cluster.
  - Scopri come configurare il Pod Identity Agent.

- Il componente aggiuntivo ha un'associazione di identità del pod valida.
  - Utilizzate il AWS CLI per recuperare le associazioni per il nome dell'account di servizio utilizzato dal componente aggiuntivo.

```
aws eks list-pod-identity-associations --cluster-name <cluster-name>
```

- Il ruolo IAM previsto ha la politica di fiducia richiesta per EKS Pod Identities.
  - Utilizza il AWS CLI per recuperare la politica di attendibilità per un componente aggiuntivo.

```
aws iam get-role --role-name <role-name> --query Role.AssumeRolePolicyDocument
```

- Il ruolo IAM previsto dispone delle autorizzazioni necessarie per il componente aggiuntivo.
  - Utilizzalo AWS CloudTrail per rivedere i nostri `AccessDenied` eventi`UnauthorizedOperation`.
- Il nome dell'account di servizio nell'associazione di identità del pod corrisponde al nome dell'account di servizio utilizzato dal componente aggiuntivo.
  - Consulta la documentazione del componente aggiuntivo per determinare il nome dell'account del servizio.

Verifica dell'immagine di container durante l'implementazione

Se utilizzi AWS Signer e desideri verificare le immagini dei container firmate al momento dell'implementazione, puoi utilizzare una delle seguenti soluzioni:

- Gatekeeper e Ratify: usa Gatekeeper come controller di ammissione e Ratify configurato con un plug-in AWS Signer come Web hook per la convalida delle firme.
- Kyverno: un motore di policy Kubernetes configurato con un plug-in AWS Signer per la convalida delle firme.

> **Note**
>
> Prima di verificare le firme delle immagini del container, configura l'archivio di attendibilità e la politica di attendibilità di Notation, come richiesto dal controller di ammissione selezionato.

Formazione del Machine learning utilizzando Elastic Fabric Adapter

In questa sezione viene descritto come integrare Elastic Fabric Adapter (EFA) con i Pods implementati nel cluster Amazon EKS. Elastic Fabric Adapter (EFA) è un'interfaccia di rete per le istanze Amazon EC2 che consente di eseguire applicazioni che richiedono livelli elevati di comunicazioni internodale su larga scala in AWS. La sua interfaccia hardware di bypass del sistema operativo personalizzata migliora le prestazioni delle comunicazioni tra istanze, che è fondamentale per dimensionare queste applicazioni. Con EFA, le applicazioni High Performance Computing (HPC) che utilizzano le applicazioni MPI (Message Passing Interface) e Machine Learning (ML) e che utilizzano NVIDIA Collective Communications Library (NCCL) possono dimensionare fino a migliaia di CPU o GPU. Di conseguenza, si ottengono le prestazioni applicative dei cluster HPC locali con l'elasticità e la flessibilità on-demand del cloud. AWS L'integrazione di EFA con le applicazioni in esecuzione su cluster Amazon EKS può ridurre il tempo necessario per completare carichi di lavoro di formazione distribuiti su larga scala senza dover aggiungere ulteriori istanze al cluster. Per ulteriori informazioni su EFA, consulta Elastic Fabric Adapter.

Il plugin EFA descritto in questo argomento supporta pienamente istanze Amazon EC2 P4d, che rappresentano l'attuale stato dell'arte nel machine learning distribuito nel cloud. Ciascun istanza `p4d.24xlarge` ha otto GPU NVIDIA A100 e GPUDirectRDMA a 400 Gbps su EFA. GPUDirectRDMA consente di avere una comunicazione diretta da GPU a GPU tra i nodi con bypass della CPU, aumentando la larghezza di banda di comunicazione collettiva e riducendo la latenza. L'integrazione di Amazon EKS e EFA con istanze P4d offre un metodo ottimale per sfruttare l'istanza di elaborazione Amazon EC2 dalle prestazioni più elevate per l'addestramento del machine learning distribuito.

Prerequisiti

- Un cluster Amazon EKS esistente. Se non disponi di un cluster esistente, consulta una delle guide Guida introduttiva ad Amazon EKS per crearne uno. Il cluster deve essere implementato in un VPC con almeno una sottorete privata con indirizzi IP disponibili sufficienti in cui implementare i nodi. La sottorete privata deve disporre di un accesso Internet in uscita fornito da un dispositivo esterno, ad esempio un gateway NAT.

  Se prevedi di utilizzare `eksctl` per creare il gruppo di nodi, `eksctl` può anche creare un cluster per tuo conto.
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o. AWS CloudShell Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi

di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella home directory nella Guida AWS CLI per l'AWS CloudShell utente.

- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Prima di avviare i nodi worker che supportano più di un Elastic Fabric Adapter, come ad esempio `p4d.24xlarge`, è necessario disporre della versione `1.7.10` di Amazon VPC CNI plugin for Kubernetes. Per ulteriori informazioni sull'aggiornamento della versione del Amazon VPC CNI plugin for Kubernetes, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS.

Creazione di un gruppo di nodi

La procedura seguente consente di creare un gruppo di nodi con un gruppo di nodi supportato da `p4d.24xlarge` con interfacce EFA e GPUDirect RDMA, e di eseguire un test di esempio NVIDIA Collective Communications Library (NCCL) per prestazioni NCCL multi-nodo utilizzando più EFA. L'esempio può essere utilizzato come modello per l'addestramento del deep learning distribuito su Amazon EKS utilizzando più EFA.

1. Determina quali tipi di istanze Amazon EC2 che supportano EFA sono disponibili nei Regione AWS nodi in cui desideri implementare. Sostituiscilo *`region-code`* con Regione AWS quello in cui desideri distribuire il tuo gruppo di nodi.

   ```
   aws ec2 describe-instance-types --region region-code --filters Name=network-
   info.efa-supported,Values=true \
       --query "InstanceTypes[*].[InstanceType]" --output text
   ```

   Quando distribuisci i nodi, il tipo di istanza che desideri distribuire deve essere disponibile nel cluster in Regione AWS cui si trova il cluster.

2. Determina in quali zone di disponibilità è disponibile il tipo di istanza che desideri implementare. In questo tutorial, viene utilizzato il tipo di `p4d.24xlarge` istanza Regione AWS che deve essere restituito nell'output per quanto specificato nel passaggio precedente. Quando distribuisci i nodi in un cluster di produzione, sostituiscili *`p4d.24xlarge`* con qualsiasi tipo di istanza restituito nel passaggio precedente.

```
aws ec2 describe-instance-type-offerings --region region-code --location-type
 availability-zone --filters Name=instance-type,Values=p4d.24xlarge \
    --query 'InstanceTypeOfferings[*].Location' --output text
```

Di seguito viene riportato un output di esempio:

```
us-west-2a    us-west-2c    us-west-2b
```

Prendi nota delle zone di disponibilità restituite per l'uso nelle fasi successive. Quando implementi i nodi in un cluster, il tuo VPC deve disporre di sottoreti con indirizzi IP disponibili in una delle zone di disponibilità restituite nell'output.

3. Crea un gruppo di nodi utilizzando uno dei due `eksctl` o il comando AWS CLI e AWS CloudFormation.

eksctl

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

1. Copia i contenuti seguenti in un file denominato *`efa-cluster.yaml`*. Sostituisci i *`example values`* con i valori in tuo possesso. Puoi sostituire *`p4d.24xlarge`* con un'istanza diversa, ma in questo caso assicurati che i valori per `availabilityZones` corrispondano a zone di disponibilità restituite per il tipo di istanza nel passaggio 1.

```
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-efa-cluster
  region: region-code
```

```
  version: "1.XX"

iam:
  withOIDC: true

availabilityZones: ["us-west-2a", "us-west-2c"]

managedNodeGroups:
  - name: my-efa-ng
    instanceType: p4d.24xlarge
    minSize: 1
    desiredCapacity: 2
    maxSize: 3
    availabilityZones: ["us-west-2a"]
    volumeSize: 300
    privateNetworking: true
    efaEnabled: true
```

2. Crea un gruppo di nodi gestito in un cluster esistente.

```
eksctl create nodegroup -f efa-cluster.yaml
```

Se non disponi di un cluster esistente, puoi eseguire il comando seguente per creare un cluster e il gruppo di nodi.

```
eksctl create cluster -f efa-cluster.yaml
```

> **Note**
>
> Poiché il tipo di istanza utilizzato in questo esempio ha delle GPU, `eksctl` installa automaticamente il plugin del dispositivo NVIDIA Kubernetes su ogni istanza per tuo conto.

AWS CLI and AWS CloudFormation

Esistono diversi requisiti per le reti EFA, tra cui la creazione di un gruppo di sicurezza specifico EFA, la creazione di un gruppo di collocamento Amazon EC2 e la creazione di un modello di avvio che specifica una o più interfacce EFA e include l'installazione dei driver EFA nell'ambito dei dati utente Amazon EC2. Per ulteriori informazioni sui requisiti

EFA, consulta Get started with EFA and MPI nella Amazon EC2 User Guide. Le seguenti fasi creano tutto ciò per tuo conto. Sostituisci i *valori di esempio* con i valori in tuo possesso.

1. Imposta alcune variabili utilizzate nelle fasi successive. Sostituisci tutti i *example values* con i valori in tuo possesso. Sostituisci tutte le stringhe *my-cluster* con il nome del cluster esistente. Il valore for `node_group_resources_name` viene successivamente utilizzato per creare uno stack. AWS CloudFormation Il valore per `node_group_name` viene utilizzato in seguito per creare il gruppo di nodi nel cluster.

   ```
   cluster_name="my-cluster"
   cluster_region="region-code"
   node_group_resources_name="my-efa-nodegroup-resources"
   node_group_name="my-efa-nodegroup"
   ```

2. Identificare una sottorete privata nel VPC, e verificare che si trovi e sia disponibile nella stessa zona di disponibilità in cui si desidera implementare il tipo di istanza.

   a. Recuperare la versione del cluster e memorizzarla in una variabile da utilizzare in un passaggio successivo.

      ```
      cluster_version=$(aws eks describe-cluster \
          --name $cluster_name \
          --query "cluster.version" \
          --output text)
      ```

   b. Recuperare l'ID VPC in cui si trova il cluster e memorizzarlo in una variabile per utilizzarlo in un passaggio successivo.

      ```
      vpc_id=$(aws eks describe-cluster \
          --name $cluster_name \
          --query "cluster.resourcesVpcConfig.vpcId" \
          --output text)
      ```

   c. Recuperare l'ID del gruppo di sicurezza del piano di controllo per il cluster e memorizzarlo in una variabile da utilizzare in un passaggio successivo.

      ```
      control_plane_security_group=$(aws eks describe-cluster \
          --name $cluster_name \
          --query "cluster.resourcesVpcConfig.clusterSecurityGroupId" \
          --output text)
      ```

d. Ottenere l'elenco degli ID sottorete nel VPC che si trovano in una zona di disponibilità restituita al passaggio 1.

```
aws ec2 describe-subnets \
    --filters "Name=vpc-id,Values=$vpc_id" "Name=availability-
zone,Values=us-west-2a" \
    --query 'Subnets[*].SubnetId' \
    --output text
```

Se non viene restituito alcun output, provare con una zona di disponibilità diversa nel passaggio 1. Se nessuna delle sottoreti si trova in una zona di disponibilità restituita nella fase 1, è necessario creare una sottorete in una zona di disponibilità restituita nella fase 1. Se il VPC non dispone di spazio per creare un'altra sottorete,ìpuoi aggiungere un blocco CIDR al VPC e creare sottoreti nel nuovo blocco CIDR o creare un nuovo cluster in un nuovo VPC.

e. Controllando la tabella di routing della sottorete, verifica se la sottorete è privata.

```
aws ec2 describe-route-tables \
    --filter Name=association.subnet-id,Values=subnet-0d403852a65210a29 \
    --query "RouteTables[].Routes[].GatewayId" \
    --output text
```

Di seguito viene riportato un output di esempio:

```
local
```

Se l'output è `local igw-02adc64c1b72722e2`, la sottorete è pubblica. È necessario selezionare una sottorete privata in una zona di disponibilità restituita al passaggio 1. Dopo aver identificato una sottorete privata, annotare il relativo ID da utilizzare in un passaggio successivo.

f. Impostare una variabile con l'ID della sottorete privata del passaggio precedente per utilizzarla nei passaggi successivi.

```
subnet_id=your-subnet-id
```

3. Scarica il AWS CloudFormation modello.

```
curl -O https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/
cloudformation/efa-p4d-managed-nodegroup.yaml
```

4. Copia il testo seguente sul computer. Sostituisci *p4d.24xlarge* con un tipo di istanza dal passaggio 1. Sostituisci *subnet-0d403852a65210a29* con l'ID della sottorete privata identificata nella fase 2.b.v. Sostituisci *path-to-downloaded-cfn-template* con il percorso per il `efa-p4d-managed-nodegroup.yaml` scaricato nella fase precedente. Sostituisci *your-public-key-name* con il nome della chiave pubblica. Dopo aver effettuato le sostituzioni, esegui il comando modificato.

```
aws cloudformation create-stack \
    --stack-name ${node_group_resources_name} \
    --capabilities CAPABILITY_IAM \
    --template-body file://path-to-downloaded-cfn-template \
    --parameters \
       ParameterKey=ClusterName,ParameterValue=${cluster_name} \
       ParameterKey=ClusterControlPlaneSecurityGroup,ParameterValue=
${control_plane_security_group} \
       ParameterKey=VpcId,ParameterValue=${vpc_id} \
       ParameterKey=SubnetId,ParameterValue=${subnet_id} \
       ParameterKey=NodeGroupName,ParameterValue=${node_group_name} \
       ParameterKey=NodeImageIdSSMParam,ParameterValue=/aws/service/eks/
optimized-ami/${cluster_version}/amazon-linux-2-gpu/recommended/image_id \
       ParameterKey=KeyName,ParameterValue=your-public-key-name \
       ParameterKey=NodeInstanceType,ParameterValue=p4d.24xlarge
```

5. Determinare il momento in cui la pila implementata nel passaggio precedente viene implementato.

```
aws cloudformation wait stack-create-complete --stack-name
 $node_group_resources_name
```

   Non c'è output dal comando precedente, ma il prompt della shell non viene restituito fino a quando non viene creara la pila.

6. Creare il gruppo di nodi utilizzando le risorse create dalla pila AWS CloudFormation nel passaggio precedente.

   a. Recupera le informazioni dallo AWS CloudFormation stack distribuito e memorizzale in variabili.

```
node_instance_role=$(aws cloudformation describe-stacks \
    --stack-name $node_group_resources_name \
    --query='Stacks[].Outputs[?OutputKey==`NodeInstanceRole`].OutputValue'
 \
    --output text)
launch_template=$(aws cloudformation describe-stacks \
    --stack-name $node_group_resources_name \
    --query='Stacks[].Outputs[?OutputKey==`LaunchTemplateID`].OutputValue'
 \
    --output text)
```

b. Creare un gruppo di nodi gestiti che utilizzi il modello di avvio e il ruolo IAM del nodo creati nel passaggio precedente.

```
aws eks create-nodegroup \
    --cluster-name $cluster_name \
    --nodegroup-name $node_group_name \
    --node-role $node_instance_role \
    --subnets $subnet_id \
    --launch-template id=$launch_template,version=1
```

c. Confermare che i nodi siano stati creati.

```
aws eks describe-nodegroup \
   --cluster-name ${cluster_name} \
   --nodegroup-name ${node_group_name} | jq -r .nodegroup.status
```

Non continuare finché lo stato restituito dal comando precedente non è `ACTIVE`. Possono essere necessari diversi minuti prima che i nodi siano pronti.

7. Se si sceglie un tipo di istanza GPU, è necessario implementare il[Plugin per dispositivi NVIDIA perKubernetes](). Sostituisci *vX.X.X* con la versione [NVIDIA/k8s-device-plugin]() desiderata prima di eseguire il comando seguente.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-
plugin/vX.X.X/nvidia-device-plugin.yml
```

4. Implementa il plugin per dispositivi EFA Kubernetes.

Il plugin per dispositivi EFA Kubernetes rileva e pubblicizza le interfacce EFA come risorse allocabili a Kubernetes. Un'applicazione può consumare il tipo di risorsa estesa

`vpc.amazonaws.com/efa` in una specifica di richiesta Pod proprio come CPU e memoria. Per ulteriori informazioni, consulta Manage Huge Pages (Utilizzo di risorse estese) nella documentazione di Kubernetes. Una volta effettuata la richiesta, il plugin assegna e monta in automatico un'interfaccia EFA sul Pod. L'utilizzo del plugin per dispositivi semplifica la configurazione EFA e non richiede l'esecuzione di un Pod in modalità privilegiata.

```
helm repo add eks https://aws.github.io/eks-chart
helm install aws-efa-k8s-device-plugin --namespace kube-system eks/aws-efa-k8s-device-plugin
```

(Facoltativo) Implementazione di un'applicazione compatibile con EFA di esempio

Implementare l'operatore MPI Kubeflow

Per i test NCCL è possibile applicare l'operatore Kubeflow MPI. L'operatore MPI semplifica l'esecuzione della formazione distribuita in stile Allreduce su Kubernetes. Per ulteriori informazioni, consulta Operatore MPI su GitHub.

```
kubectl apply -f https://raw.githubusercontent.com/kubeflow/mpi-operator/master/deploy/v2beta1/mpi-operator.yaml
```

Esecuzione del test delle prestazioni NCCL multi-nodo per verificare GPUDirectrRDMA/EFA

Per verificare le prestazioni NCCL con GPUDirectRDMA su EFA, esegui il test delle prestazioni NCCL standard. Per ulteriori informazioni, consulta il repository Test NCCL ufficiale su GitHub. È possibile utilizzare il Dockerfile di esempio fornito con questo test già creato per NVIDIA CUDA `11.2` e per l'ultima versione di EFA.

In alternativa, puoi scaricare un' AWS Dockerimmagine disponibile da un repository Amazon ECR.

> **Important**
>
> Un importante aspetto da considerare per l'adozione di EFA con Kubernetes è la configurazione e la gestione di Huge Pages come risorsa nel cluster. Per ulteriori informazioni, consulta Manage Huge Pages (Gestione di Huge Pages) nella documentazione di Kubernetes. Le istanze Amazon EC2 con il driver EFA installato allocano in via anticipata

5.128 Huge Pages da 2M, che possono essere richieste come risorse da utilizzare nelle specifiche del processo.

Completa i seguenti passaggi per eseguire un test delle prestazioni NCCL a due nodi. Nel processo di test NCCL di esempio, ogni worker richiede otto GPU, 5210Mi di HugePages-2Mi, quattro EFA e 8000Mi di memoria, il che significa che ogni worker consuma tutte le risorse di un'istanza `p4d.24xlarge`.

1. Crea il processo di test NCCL.

```
kubectl apply -f https://raw.githubusercontent.com/aws-samples/aws-efa-eks/main/examples/simple/nccl-efa-tests.yaml
```

   Di seguito viene riportato un output di esempio:

   nccl-tests-efa mpijob.kubeflow.org/ creato

2. Visualizza i Pods in esecuzione.

```
kubectl get pods
```

   Di seguito viene riportato un output di esempio:

```
NAME                              READY   STATUS     RESTARTS   AGE
nccl-tests-efa-launcher-nbql9  0/1     Init:0/1   0          2m49s
nccl-tests-efa-worker-0          1/1     Running    0          2m49s
nccl-tests-efa-worker-1          1/1     Running    0          2m49s
```

   L'operatore MPI crea un Pod di avvio e 2 Pods worker (uno su ciascun nodo).

3. Visualizza il log per il Pod `efa-launcher`. Sostituisci *`wzr8j`* con il valore dell'output.

```
kubectl logs -f nccl-tests-efa-launcher-nbql9
```

Per altri esempi, consulta il repository Esempi di EFA per Amazon EKS su GitHub.

Inferenza del machine learning utilizzando AWS Inferentia

In questo argomento viene descritto come creare un cluster Amazon EKS con nodi che eseguono istanze Amazon EC2 Inf1 e (facoltativo) come implementare un'applicazione di esempio. Le istanze Amazon EC2 Inf1 sono alimentate da chip AWS Inferentia, creati su misura AWS per fornire inferenze ad alte prestazioni e al minor costo nel cloud. I modelli di apprendimento automatico vengono distribuiti nei contenitori utilizzando AWS Neuron, un kit di sviluppo software (SDK) specializzato composto da un compilatore, un runtime e strumenti di profilazione che ottimizzano le prestazioni di inferenza dell'apprendimento automatico dei chip Inferentia. AWS Neuron supporta i più diffusi framework di machine learning come TensorFlow, PyTorch e MXNet.

> **Note**
>
> Gli ID logici dei dispositivi neuronali devono essere contigui. Se un Pod che richiede più dispositivi Neuron è pianificato su un tipo di istanza `inf1.6xlarge` o `inf1.24xlarge` (che ha più di un dispositivo Neuron), il Pod non si avvia se lo scheduler Kubernetes seleziona ID dispositivo non contigui. Per ulteriori informazioni, consulta Gli ID logici del dispositivo devono essere contigui su GitHub.

Prerequisiti

- Avere `eksctl` installato sul computer. Se non è installato, consulta Installation nella documentazione di `eksctl`.
- Avere `kubectl` installato sul computer. Per ulteriori informazioni, consulta Installazione o aggiornamento di `kubectl`.
- (Facoltativo) Avere `python3` installato sul computer. Se non lo hai installato, consultare i download di Python per le istruzioni di installazione.

Creazione di un cluster

Per creare un cluster Amazon EKS con nodi di istanze Amazon EC2 Inf1

1. Creare un cluster con nodi di istanze Amazon EC2 Inf1. Puoi sostituire *`inf1.2xlarge`* con qualsiasi tipo di istanza Inf1. L'utility `eksctl` rileva l'avvio di un gruppo di nodi con un tipo di istanza `Inf1` e avvia i nodi usando una delle AMI Amazon Linux accelerate ottimizzate per Amazon EKS.

> **Note**
>
> Non è possibile utilizzare i ruoli IAM per gli account di servizio con Serving. TensorFlow

```
eksctl create cluster \
    --name inferentia \
    --region region-code \
    --nodegroup-name ng-inf1 \
    --node-type inf1.2xlarge \
    --nodes 2 \
    --nodes-min 1 \
    --nodes-max 4 \
    --ssh-access \
    --ssh-public-key your-key \
    --with-oidc
```

> **Note**
>
> Notare il valore della seguente riga dell'output. Viene utilizzato in un passaggio successivo (facoltativo).
>
> ```
> [9]  adding identity "arn:aws:iam::111122223333:role/
> eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09" to auth
>  ConfigMap
> ```

Quando si avvia un gruppo di nodi con `Inf1` istanze, installa `eksctl` automaticamente il plug-in del dispositivo AWS Kubernetes Neuron. Questo plug-in pubblicizza i dispositivi Neuron come risorsa di sistema allo scheduler Kubernetes, che può essere richiesto da un container. Oltre alle policy IAM dei nodi Amazon EKS di default, viene aggiunta la policy di accesso di sola lettura Amazon S3 in modo che l'applicazione di esempio, che verrà trattata in un passaggio successivo, possa caricare un modello addestrato da Amazon S3.

2. Assicurati che tutti i Pods siano stati avviati correttamente.

```
kubectl get pods -n kube-system
```

Output abbreviato:

```
NAME                                   READY   STATUS    RESTARTS   AGE
[...]
neuron-device-plugin-daemonset-6djhp   1/1     Running   0          5m
neuron-device-plugin-daemonset-hwjsj   1/1     Running   0          5m
```

(Facoltativo) Implementate un'immagine dell'applicazione Serving TensorFlow

Un modello addestrato deve essere compilato in un target Inferentia prima di poter essere implementato nelle istanze Inferentia. Per continuare, avrai bisogno di un TensorFlow modello ottimizzato per Neuron salvato in Amazon S3. Se non ne hai già uno SavedModel, segui il tutorial per creare un modello ResNet 50 compatibile con Neuron e carica il risultato SavedModel su S3. ResNet-50 è un popolare modello di apprendimento automatico utilizzato per attività di riconoscimento delle immagini. Per ulteriori informazioni sulla compilazione dei modelli Neuron, vedere The AWS Inferentia Chip With DLAMI nella Developer Guide. AWS Deep Learning AMI

Il manifesto di distribuzione di esempio gestisce un contenitore di servizi di inferenza precostruito TensorFlow fornito da AWS Deep Learning Containers. All'interno del contenitore si trovano AWS Neuron Runtime e l' TensorFlow applicazione Serving. Un elenco completo di container deep learning pre-costruiti ottimizzati per Neuron è consultabile su GitHub in Immagini disponibili. All'avvio, il DLC recupererà il tuo modello da Amazon S3, avvierà Neuron TensorFlow Serving con il modello salvato e aspetterà le richieste di previsione.

Il numero di dispositivi Neuron assegnati all'applicazione di servizio può essere regolato modificando la risorsa `aws.amazon.com/neuron` nella implementazione yaml. Tieni presente che la comunicazione tra TensorFlow Serving e il runtime Neuron avviene tramite GRPC, il che richiede il trasferimento della funzionalità al contenitore. `IPC_LOCK`

1. Aggiungere la policy `AmazonS3ReadOnlyAccess` IAM al ruolo di istanza del nodo creato nel passaggio 1 di Creazione di un cluster. Ciò è necessario affinché l'applicazione di esempio possa caricare un modello formato da Amazon S3.

   ```
   aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess \
       --role-name eksctl-inferentia-nodegroup-ng-in-NodeInstanceRole-FI7HIYS3BS09
   ```

2. Crea un file denominato `rn50_deployment.yaml` con i seguenti contenuti. Aggiornare il codice di Regione e il percorso del modello in modo che corrispondano alle impostazioni desiderate. Il nome del modello serve a scopo di identificazione quando un client effettua una richiesta al TensorFlow server. Questo esempio utilizza un nome di modello che corrisponde a uno script client di esempio di ResNet 50 client che verrà utilizzato in un passaggio successivo per l'invio di richieste di previsione.

```
aws ecr list-images --repository-name neuron-rtd --registry-id 790709498068 --
region us-west-2
```

```
kind: Deployment
apiVersion: apps/v1
metadata:
  name: eks-neuron-test
  labels:
    app: eks-neuron-test
    role: master
spec:
  replicas: 2
  selector:
    matchLabels:
      app: eks-neuron-test
      role: master
  template:
    metadata:
      labels:
        app: eks-neuron-test
        role: master
    spec:
      containers:
        - name: eks-neuron-test
          image: 763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-inference-
neuron:1.15.4-neuron-py37-ubuntu18.04
          command:
            - /usr/local/bin/entrypoint.sh
          args:
            - --port=8500
            - --rest_api_port=9000
            - --model_name=resnet50_neuron
            - --model_base_path=s3://your-bucket-of-models/resnet50_neuron/
          ports:
            - containerPort: 8500
```

```
            - containerPort: 9000
          imagePullPolicy: IfNotPresent
          env:
            - name: AWS_REGION
              value: "us-east-1"
            - name: S3_USE_HTTPS
              value: "1"
            - name: S3_VERIFY_SSL
              value: "0"
            - name: S3_ENDPOINT
              value: s3.us-east-1.amazonaws.com
            - name: AWS_LOG_LEVEL
              value: "3"
          resources:
            limits:
              cpu: 4
              memory: 4Gi
              aws.amazon.com/neuron: 1
            requests:
              cpu: "1"
              memory: 1Gi
          securityContext:
            capabilities:
              add:
                - IPC_LOCK
```

3. Implementare il modello.

```
kubectl apply -f rn50_deployment.yaml
```

4. Crea un file denominato `rn50_service.yaml` con i seguenti contenuti. Le porte HTTP e gRPC vengono aperte per accettare le richieste di previsione.

```
kind: Service
apiVersion: v1
metadata:
  name: eks-neuron-test
  labels:
    app: eks-neuron-test
spec:
  type: ClusterIP
  ports:
    - name: http-tf-serving
```

```
      port: 8500
      targetPort: 8500
    - name: grpc-tf-serving
      port: 9000
      targetPort: 9000
  selector:
    app: eks-neuron-test
    role: master
```

5. Crea un Kubernetes servizio per la tua applicazione TensorFlow Model Serving.

```
kubectl apply -f rn50_service.yaml
```

(Facoltativo) Fai previsioni sul tuo TensorFlow servizio Serving

1. Per testare localmente, inoltrare la porta gRPC al servizio `eks-neuron-test`.

```
kubectl port-forward service/eks-neuron-test 8500:8500 &
```

2. Creare uno script Python chiamato `tensorflow-model-server-infer.py` con il seguente contenuto. Questo script esegue inferenza tramite gRPC, che è framework di servizio.

```
import numpy as np
   import grpc
   import tensorflow as tf
   from tensorflow.keras.preprocessing import image
   from tensorflow.keras.applications.resnet50 import preprocess_input
   from tensorflow_serving.apis import predict_pb2
   from tensorflow_serving.apis import prediction_service_pb2_grpc
   from tensorflow.keras.applications.resnet50 import decode_predictions

   if __name__ == '__main__':
       channel = grpc.insecure_channel('localhost:8500')
       stub = prediction_service_pb2_grpc.PredictionServiceStub(channel)
       img_file = tf.keras.utils.get_file(
           "./kitten_small.jpg",
           "https://raw.githubusercontent.com/awslabs/mxnet-model-server/master/
docs/images/kitten_small.jpg")
       img = image.load_img(img_file, target_size=(224, 224))
       img_array = preprocess_input(image.img_to_array(img)[None, ...])
       request = predict_pb2.PredictRequest()
```

```
        request.model_spec.name = 'resnet50_inf1'
        request.inputs['input'].CopyFrom(
            tf.make_tensor_proto(img_array, shape=img_array.shape))
        result = stub.Predict(request)
        prediction = tf.make_ndarray(result.outputs['output'])
        print(decode_predictions(prediction))
```

3. Eseguire lo script per inviare previsioni al servizio.

```
python3 tensorflow-model-server-infer.py
```

Di seguito viene riportato un output di esempio:

```
[[(u'n02123045', u'tabby', 0.68817204), (u'n02127052', u'lynx', 0.12701613),
 (u'n02123159', u'tiger_cat', 0.08736559), (u'n02124075', u'Egyptian_cat',
 0.063844085), (u'n02128757', u'snow_leopard', 0.009240591)]]
```

Gestione dei cluster

Questo capitolo include i seguenti argomenti per aiutarti a gestire il cluster. Puoi visualizzare informazioni relative alle risorse Kubernetes anche tramite la AWS Management Console.

- La dashboard Kubernetes è un'interfaccia utente generica basata sul Web per i cluster Kubernetes. Consente agli utenti di gestire le applicazioni in esecuzione nel cluster e risolverne i problemi, nonché di gestire il cluster stesso. Per ulteriori informazioni, consulta il repository GitHub della dashboard Kubernetes.
- Installazione di Kubernetes Metrics Server: Kubernetes Metrics Server è un aggregatore dei dati di utilizzo delle risorse nel cluster. Non viene implementato per impostazione predefinita nel cluster, ma viene utilizzato dai componenti aggiuntivi di Kubernetes, ad esempio la dashboard Kubernetes e Horizontal Pod Autoscaler. In questo argomento viene descritto come installare il server dei parametri.
- Utilizzo di Helm con Amazon EKS: il programma di gestione del pacchetto Helm per Kubernetes consente di installare e gestire le applicazioni sul cluster Kubernetes. Questo argomento consente di installare ed eseguire i file binari Helm per consentire l'installazione e la gestione di grafici utilizzando la CLI Helm nel computer.
- Assegnazione di tag alle risorse Amazon EKS: per semplificare la gestione delle risorse Amazon EKS, è possibile assegnare metadati personalizzati a ciascuna risorsa sotto forma di tag. Questo argomento descrive i tag e mostra come crearli.
- Service Quotas di Amazon EKS: l'account AWS dispone delle seguenti quote di default, precedentemente definite limiti, per ogni servizio AWS. Scopri le quote per Amazon EKS e come aumentarle.

Monitoraggio dei costi

Il monitoraggio dei costi è un aspetto essenziale della gestione dei Kubernetes cluster su Amazon EKS. Acquisendo visibilità sui costi dei cluster, puoi ottimizzare l'utilizzo delle risorse, fissare i budget e prendere decisioni basate sui dati sulle tue implementazioni. Amazon EKS offre due soluzioni di monitoraggio dei costi, ognuna con vantaggi unici, per aiutarti a tracciare e allocare i costi in modo efficace:

AWS Dati di allocazione dei costi suddivisi per Amazon EKS: questa funzionalità nativa si integra perfettamente con la Console di AWS fatturazione, consentendoti di analizzare e allocare i costi

utilizzando la stessa interfaccia e gli stessi flussi di lavoro familiari utilizzati per altri servizi. AWS Con l'allocazione suddivisa dei costi, puoi ottenere informazioni dettagliate sui Kubernetes costi direttamente insieme alle altre AWS spese, semplificando l'ottimizzazione olistica dei costi in tutto l'ambiente. AWS Puoi anche sfruttare le funzionalità di AWS fatturazione esistenti come Cost Categories e Cost Anomaly Detection per migliorare ulteriormente le tue capacità di gestione dei costi. Per ulteriori informazioni, consulta Comprensione dei dati di allocazione dei costi suddivisi nella Billing User Guide AWS .

Kubecost— Amazon EKS supporta Kubecost, uno strumento di monitoraggio dei costi di Kubernetes. Kubecost offre un approccio ricco di funzionalità e nativo di Kubernetes al monitoraggio dei costi, che fornisce analisi granulari dei costi in base alle risorse Kubernetes, consigli per l'ottimizzazione dei costi, dashboard e report. out-of-the-box Kubecost recupera anche dati accurati sui prezzi integrandosi con il rapporto sui AWS costi e sull'utilizzo, assicurandoti una visione precisa dei costi di Amazon EKS. Scopri come installare. Kubecost

AWS Fatturazione: suddivisione dei costi

Monitoraggio dei costi tramite dati di allocazione dei costi AWS suddivisi per Amazon EKS

Puoi utilizzare i dati di allocazione dei costi AWS suddivisi per Amazon EKS per ottenere una visibilità granulare dei costi per i tuoi cluster Amazon EKS. Ciò ti consente di analizzare, ottimizzare e riaddebitare i costi e l'utilizzo delle tue applicazioni. Kubernetes Allochi i costi delle applicazioni a singole unità aziendali e team in base alla CPU e alle risorse di memoria di Amazon EC2 utilizzate dalla Kubernetes tua applicazione. I dati di allocazione dei costi suddivisi per Amazon EKS offrono visibilità sul costo per Pod e consentono di aggregare i dati sui costi per Pod utilizzando namespace, cluster e altre primitive. Kubernetes Di seguito sono riportati alcuni esempi di Kubernetes primitive che puoi utilizzare per analizzare i dati di allocazione dei costi di Amazon EKS.

- Nome cluster
- Implementazione
- Spazio dei nomi
- Nodo
- Nome del carico di lavoro
- Tipo di carico di lavoro

Per ulteriori informazioni sull'utilizzo dei dati di allocazione dei costi suddivisi, consulta Comprendere i dati di allocazione dei costi suddivisi nella Billing User Guide. AWS

Configura i report sui costi e sull'utilizzo

Puoi attivare la suddivisione dei dati di allocazione dei costi per EKS nella console di gestione dei costi o negli AWS SDK. AWS Command Line Interface

Utilizza quanto segue per suddividere i dati di allocazione dei costi:

1. Attiva la suddivisione dei dati di allocazione dei costi. Per ulteriori informazioni, vedere Abilitazione dei dati di allocazione dei costi suddivisi nella Guida per l' AWS Cost and Usage Report utente.
2. Includi i dati in un rapporto nuovo o esistente.
3. Visualizza il rapporto. Per visualizzare i file di report in Amazon Simple Storage Service, puoi utilizzare la console Gestione costi e fatturazione.

Kubecost

Amazon EKS supporta Kubecost, che puoi utilizzare per monitorare i costi suddivisi per risorse Kubernetes tra cui Pods, nodi, spazi dei nomi ed etichette. Come amministratore della piattaforma Kubernetes e leader finanziario, puoi usare Kubecost per visualizzare una scomposizione degli addebiti di Amazon EKS, allocare i costi e riaddebitare unità organizzative come i team applicativi. È possibile fornire ai team interni e alle unità aziendali dati sui costi trasparenti e accurati basati sulla fattura effettiva AWS . Inoltre, puoi anche ottenere consigli personalizzati per l'ottimizzazione dei costi in base all'ambiente dell'infrastruttura e ai modelli di utilizzo all'interno dei cluster. Per ulteriori informazioni su Kubecost, consulta la documentazione su Kubecost.

Amazon EKS offre un pacchetto AWS ottimizzato Kubecost per la visibilità dei costi dei cluster. Puoi utilizzare i contratti di AWS supporto esistenti per ottenere assistenza.

Prerequisiti

- Un cluster Amazon EKS esistente. Per implementarne uno, consulta Guida introduttiva ad Amazon EKS. Il cluster deve avere nodi Amazon EC2 perché non è possibile eseguire Kubecost sui nodi Fargate.
- Lo strumento a riga di comando `kubectl` è installato sul dispositivo o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:

- La versione 3.9.0 di Helm o successiva deve essere configurata sul dispositivo o sul AWS CloudShell. Per installare o aggiornare Helm, consultare the section called “Utilizzo di Helm”.
- Se la versione del tuo cluster è `1.23` o successiva, devi disporre di the section called “Driver CSI per Amazon EBS” installato nel cluster.

Per installare Kubecost

1. Determina la versione di Kubecost da installare. Puoi vedere le versioni disponibili all'indirizzo kubecost/cost-analyzer nella galleria pubblica di Amazon ECR. Per ulteriori informazioni sulla compatibilità delle Kubecost versioni e di Amazon EKS, consulta i Requisiti ambientali nella documentazione di Kubecost.
2. Installa Kubecost con il comando seguente: *Sostituisci kubecost-version con il valore recuperato da ECR, ad esempio 1.108.1.*

   ```
   helm upgrade -i kubecost oci://public.ecr.aws/kubecost/cost-analyzer --
   version kubecost-version \
       --namespace kubecost --create-namespace \
       -f https://raw.githubusercontent.com/kubecost/cost-analyzer-helm-chart/develop/
   cost-analyzer/values-eks-cost-monitoring.yaml
   ```

   Kubecost rilascia regolarmente nuove versioni. Puoi aggiornare la tua versione utilizzando `helm upgrade`. Per impostazione predefinita, l'installazione include un server locale Prometheus e `kube-state-metrics`. È possibile personalizzare la distribuzione per utilizzare Amazon Managed Service for Prometheus facendo riferimento alla documentazione riportata in Integrazione con il monitoraggio dei costi di Amazon EKS. Per un elenco di tutte le altre impostazioni che puoi configurare, consulta il file di configurazione di esempio su. GitHub
3. Assicurati che i Pods richiesti siano in esecuzione.

   ```
   kubectl get pods -n kubecost
   ```

   Di seguito viene riportato un output di esempio:

   ```
   NAME                                          READY   STATUS    RESTARTS   AGE
   kubecost-cost-analyzer-b9788c99f-5vj5b        2/2     Running   0          3h27m
   kubecost-kube-state-metrics-99bb8c55b-bn2br   1/1     Running   0          3h27m
   kubecost-prometheus-server-7d9967bfc8-9c8p7   2/2     Running   0          3h27m
   ```
4. Sul dispositivo, abilita il port forwarding per esporre la dashboard Kubecost.

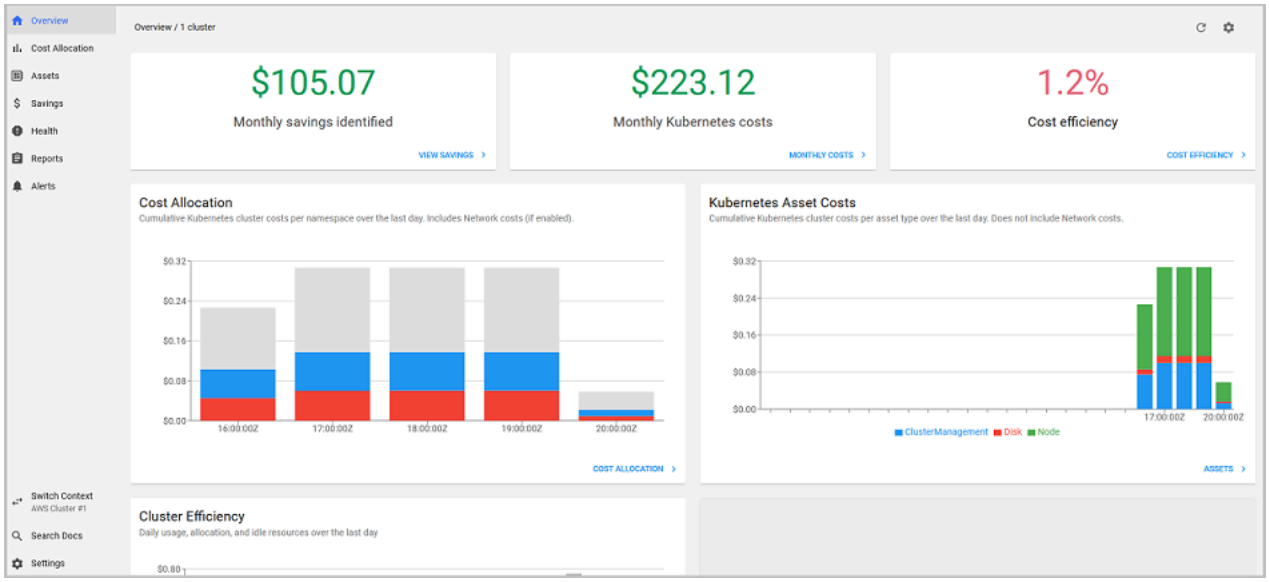
```
kubectl port-forward --namespace kubecost deployment/kubecost-cost-analyzer 9090
```

In alternativa, è possibile utilizzare AWS Load Balancer Controller per esporre Kubecost e utilizzare Amazon Cognito per autenticazione, autorizzazione e gestione degli utenti. Per ulteriori informazioni, vedi Come usare Application Load Balancer e Amazon Cognito autenticare gli utenti per le tue Kubernetes app web.

5. Sullo stesso dispositivo su cui hai completato il passaggio precedente, apri un browser Web e inserisci il seguente indirizzo.

```
http://localhost:9090
```

Vedi la pagina Panoramica Kubecost nel browser. Kubecost potrebbe impiegare dai 5 ai 10 minuti per raccogliere le metriche. Puoi visualizzare la tua spesa Amazon EKS, inclusi i costi cumulativi dei cluster, costi patrimoniali Kubernetes associati e spesa mensile aggregata.



6. Per tenere traccia dei costi a livello di cluster, tagga le tue risorse Amazon EKS per la fatturazione. Per ulteriori informazioni, consulta Tagging delle risorse per la fatturazione.

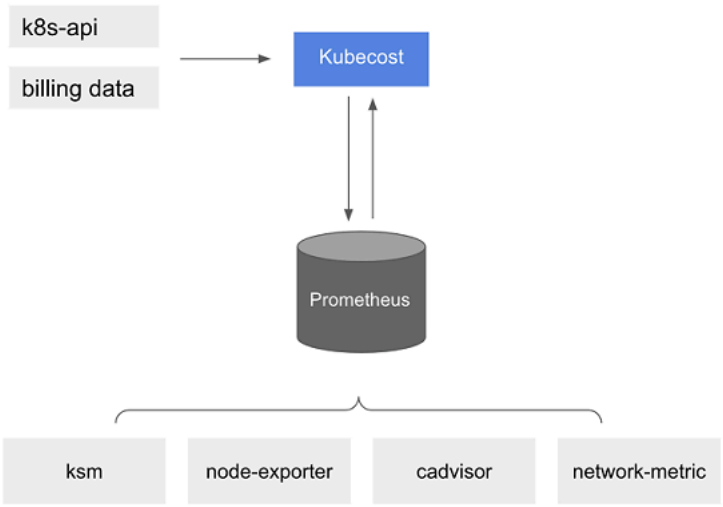
Puoi anche visualizzare le seguenti informazioni selezionandole nel riquadro sinistro della dashboard:

- Ripartizione dei costi: visualizza i costi mensili di Amazon EKS e i costi cumulativi per ciascuno dei tuoi spazi dei nomi e altre dimensioni negli ultimi sette giorni. Questo è utile per capire quali parti della tua applicazione contribuiscono alla spesa di Amazon EKS.

- Risorse: visualizza i costi delle risorse dell'infrastruttura AWS che sono associati alle risorse Amazon EKS.

Funzionalità aggiuntive

- Metriche dei costi di esportazione: il monitoraggio ottimizzato dei costi di Amazon EKS viene implementato con Kubecost e Prometheus, che è un sistema di monitoraggio open source e un database di serie temporali. Kubecost legge la metrica da Prometheus quindi esegue i calcoli di allocazione dei costi e riscrive le metriche su Prometheus. Il front-end Kubecost legge le metriche da Prometheus e le mostra sull'interfaccia utente Kubecost. Il diagramma seguente illustra tale architettura.



  Con Prometheus preinstallato, puoi scrivere interrogazioni per importare dati Kubecost nel tuo attuale sistema di business intelligence per ulteriori analisi. Puoi anche usarlo come origine dei dati per la tua attuale dashboard Grafana per visualizzare i costi dei cluster Amazon EKS che i team interni conoscono. Per saperne di più su come scrivere Prometheus query, consulta il `readme` file di Prometheusconfigurazione GitHub o usa i modelli Grafana JSON di esempio nel repository KubecostGithub come riferimenti.
- AWS Cost and Usage Report integrazione: per eseguire calcoli di allocazione dei costi per il cluster Amazon EKS, Kubecost recupera le informazioni pubbliche sui prezzi Servizi AWS e le AWS risorse dall'API AWS Price List. Puoi anche effettuare l'integrazione Kubecost con AWS Cost and Usage Reportper migliorare la precisione delle informazioni sui prezzi specifiche del tuo. Account AWS Queste informazioni includono programmi di discount aziendali, utilizzo di istanze riservate,

Savings Plans e utilizzo spot. Per ulteriori informazioni su come funziona l' AWS Cost and Usage Report integrazione, consulta AWS Cloud Billing Integration nella Kubecost documentazione.

Rimozione di Kubecost

Puoi rimuovere Kubecost dal cluster con i seguenti comandi.

```
helm uninstall kubecost --namespace kubecost
kubectl delete ns kubecost
```

Domande frequenti

Consulta le seguenti domande e risposte comuni sull'utilizzo di Kubecost con Amazon EKS.

Qual è la differenza tra il pacchetto personalizzato di Kubecost e la versione gratuita di Kubecost (noto anche come OpenCost)?

AWS e Kubecost ha collaborato per offrire una versione personalizzata di. Kubecost Questa versione include un sottoinsieme di funzionalità commerciali senza costi aggiuntivi. Consulta la tabella seguente per le funzionalità incluse nel pacchetto personalizzato di Kubecost.

Funzionalità	Piano gratuito di Kubecost	Ottimizzato per Amazon EKS Kubecostp acchetto personali zzato	Kubecost Enterprise
Distribuzione	Utente ospitato	Utente ospitato	Utente ospitato o Kubecost ospitato (SaaS)
Numero di cluster supportati	Illimitato	Illimitato	Illimitato
Database supportati	Locale Prometheus	Locale Prometheus o Amazon Managed Service for Prometheu s	Prometheus, Amazon Managed Service for Prometheus, Cortex, oppure Thanos

Funzionalità	Piano gratuito di Kubecost	Ottimizzato per Amazon EKS Kubecostp acchetto personali zzato	Kubecost Enterprise
Supporto per la conservazione dei database	15 giorni	Dati storici illimitati	Dati storici illimitati
KubecostConservazi one delle API (ETL)	15 giorni	15 giorni	Dati storici illimitati
Visibilità costo del cluster	Cluster singoli	Multicluster unificato	Multicluster unificato
Visibilità del cloud ibrido	-	Cluster Amazon EKS e Amazon EKS Anywhere	Supporto multi-cloud e cloud ibrido
Avvisi e report ricorrenti	-	Avvisi di efficienza, avvisi sul budget, avvisi di modifica della spesa e altro ancora	Avvisi di efficienza, avvisi sul budget, avvisi di modifica della spesa e altro ancora
Report salvati.	-	Report che utilizzano dati di 15 giorni	Report che utilizzano dati storici illimitati
Integrazione della fatturazione cloud	Obbligatorio per ogni singolo cluster	Supporto tariffario personalizzato per AWS (inclusi più cluster e più account)	Supporto tariffario personalizzato per AWS (inclusi più cluster e più account)
Suggerimenti per i risparmi	Informazioni su un singolo cluster	Informazioni su un singolo cluster	Informazioni su più cluster
Governance: audit	-	-	Verifica gli eventi relativi ai costi storici

Funzionalità	Piano gratuito di Kubecost	Ottimizzato per Amazon EKS Kubecostp acchetto personali zzato	Kubecost Enterprise
Supporto Single Sign-On (SSO)	-	Amazon Cognito supportato	Okta, Auth0, PingID, KeyCloak
Controllo accessi basato sui ruoli (RBAC) con SAML **2.0**	-	-	Okta, Auth0, PingID, Keycloak
Formazione e onboarding aziendali	-	-	Formazione completa eFinOps onboarding

Cos'è la funzionalità di Kubecost conservazione delle API (ETL)?

La Kubecost funzionalità ETL aggrega e organizza le metriche per evidenziare la visibilità dei costi a vari livelli di granularità (ad esempio `namespace-level`,`pod-level`, e`deployment-level`). Per il pacchetto personalizzato Kubecost, i clienti ottengono dati e approfondimenti dalle metriche degli ultimi 15 giorni.

Cos'è la funzionalità degli avvisi e dei report ricorrenti? Quali avvisi e report include?

Kubecostgli avvisi consentono ai team di ricevere aggiornamenti sulla spesa in tempo reale Kubernetes e la spesa per il cloud. I report ricorrenti consentono ai team di ricevere visualizzazioni personalizzate della cronologia Kubernetes e spese per il cloud. Entrambi sono configurabili utilizzando l'interfaccia utente Kubecost o valori Helm. Supportano la posta elettronica,Slack, eMicrosoft Teams.

Cosa includono i report salvati?

Kubecosti report salvati sono visualizzazioni predefinite delle metriche di costi ed efficienza. Includono il costo per cluster, namespace, etichetta e altro ancora.

Cos'è l'integrazione della fatturazione cloud?

L'integrazione con le API di AWS fatturazione consente di Kubecost visualizzare out-of-cluster i costi (come Amazon S3). Inoltre, consente a Kubecost di riconciliare Kubecost con i dati di

fatturazione effettivi del cluster, per tenere conto dell'utilizzo spot, dei piani di risparmio e degli sconti aziendali.

Cosa includono i consigli di risparmio?

Kubecost fornisce approfondimenti e automazione per aiutare gli utenti a ottimizzare Kubernetes infrastrutture e spese.

È previsto un addebito per questa funzionalità?

No. Puoi usare questa versione di Kubecost senza costi aggiuntivi. Se desideri Kubecost funzionalità aggiuntive non incluse in questo pacchetto, puoi acquistare una licenza aziendale Kubecost tramite o direttamente Marketplace AWS. Kubecost

Il supporto è disponibile?

Sì. Puoi aprire una richiesta di supporto con il AWS Support team all'indirizzo Contact AWS.

È necessaria una licenza per l'utilizzo delle funzionalità di Kubecost fornite dall'integrazione Amazon EKS?

No.

Posso integrarmi Kubecost con AWS Cost and Usage Report per creare report più accurati?

Sì. Puoi configurare Kubecost affinché importi dati da AWS Cost and Usage Report per ottenere una visibilità accurata dei costi, inclusi sconti, prezzi spot, prezzi delle istanze riservate e altro ancora. Per ulteriori informazioni, consulta AWS Cloud Billing Integration nella Kubecost documentazione.

Questa versione supporta la gestione dei costi dei cluster Kubernetes autogestiti su Amazon EC2?

No. Questa versione è compatibile solo con i cluster Amazon EKS.

Kubecost può monitorare i costi per Amazon EKS su AWS Fargate?

Kubecost fa tutto il possibile per mostrare la visibilità dei costi dei cluster per Amazon EKS su Fargate, ma con una precisione inferiore rispetto ad Amazon EKS su Amazon EC2. Ciò è dovuto principalmente alla differenza nella modalità di fatturazione dell'utilizzo. Con Amazon EKS su Fargate, ti vengono fatturate le risorse utilizzate. Con Amazon EKS sui nodi Amazon EC2, ti vengono fatturate le risorse allocate. Kubecost calcola il costo di un nodo Amazon EC2 in base alle specifiche del nodo, che includono CPU, RAM e archiviazione temporanea. Con Fargate, i costi vengono calcolati in base alle risorse richieste per i Pods Fargate.

Come posso ottenere aggiornamenti e nuove versioni di Kubecost?

Puoi aggiornare la tua versione Kubecost utilizzando le procedure di aggiornamento standard di Helm. Le versioni più recenti si trovano nella Galleria pubblica di Amazon ECR.

La CLI di **`kubectl-cost`** è supportata? Come si installa?

Sì. `Kubectl-cost` è uno strumento open source di Kubecost (licenza Apache 2.0) che fornisce l'accesso tramite CLI ai parametri di allocazione dei costi di Kubernetes. Per l'installazione`kubectl-cost`, vedi Installazione su. GitHub

L'interfaccia utente di Kubecost è supportata? Come vi si accede?

Kubecost fornisce un pannello di controllo Web a cui puoi accedere tramite l'inoltro alla porta di `kubectl`, un ingresso o un load balancer. Puoi anche utilizzare AWS Load Balancer Controller per esporre Kubecost e utilizzare Amazon Cognito per l'autenticazione, l'autorizzazione e la gestione degli utenti. Per ulteriori informazioni, consulta Come usare Application Load Balancer e Amazon Cognito per autenticare gli utenti delle Kubernetes tue app Web sul blog. AWS

Amazon EKS Anywhere è supportato?

No.

Installazione di Kubernetes Metrics Server

Kubernetes Metrics Server è un aggregatore dei dati di utilizzo delle risorse nel cluster e non viene implementato nei cluster Amazon EKS per impostazione predefinita. Per ulteriori informazioni, consulta Kubernetes Metrics Server su GitHub. Il Metrics Server è comunemente usato da altri componenti aggiuntivi di Kubernetes come Horizontal Pod Autoscaler o la dashboard di Kubernetes. Per ulteriori informazioni, consulta Pipeline di parametri delle risorse nella documentazione di Kubernetes. In questa sezione viene descritto come implementare Kubernetes Metrics Server sul cluster Amazon EKS.

> **Important**
>
> Le metriche sono destinate all' point-in-time analisi e non sono una fonte accurata per l'analisi storica. Non possono essere utilizzati come soluzione di monitoraggio o per altri scopi di dimensionamento non automatico. Per ulteriori informazioni sul monitoraggio, consulta la pagina Osservabilità in Amazon EKS.

Per implementare Metrics Server

1. implementare Metrics Server con il comando seguente:

   ```
   kubectl apply -f https://github.com/kubernetes-sigs/metrics-server/releases/latest/download/components.yaml
   ```

   Se si utilizza Fargate, sarà necessario modificare questo file. Nella configurazione predefinita, il server di metrica utilizza la porta 10250. Questa porta è riservata a Fargate. Sostituisci i riferimenti alla porta 10250 in components.yaml con un'altra porta, ad esempio 10251.

2. Verifica che l'implementazione di `metrics-server` esegua il numero di Pods desiderato con il comando seguente.

   ```
   kubectl get deployment metrics-server -n kube-system
   ```

   Di seguito viene riportato un output di esempio:

   ```
   NAME             READY   UP-TO-DATE   AVAILABLE   AGE
   metrics-server   1/1     1            1           6m
   ```

Utilizzo di Helm con Amazon EKS

Il programma di gestione del pacchetto Helm per Kubernetes consente di installare e gestire le applicazioni sul cluster Kubernetes. Per ulteriori informazioni, consultare la documentazione di Helm. Questo argomento consente di installare ed eseguire i file binari Helm per consentire l'installazione e la gestione di grafici utilizzando la CLI Helm nel sistema locale.

> **Important**
>
> Prima di installare grafici Helm nel cluster Amazon EKS, è necessario configurare `kubectl` per l'utilizzo di Amazon EKS. Se non hai già eseguito questa operazione, consultare Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS prima di continuare. Se il comando seguente va a buon fine per il cluster, la configurazione è corretta.
>
> ```
> kubectl get svc
> ```

Per installare i file binari Helm sul sistema locale

1. Eseguire il comando appropriato per il sistema operativo client.

   - Se si utilizza macOS con Homebrew, installare i binari con il comando seguente.

     ```
     brew install helm
     ```

   - Se utilizzi Windows con Chocolatey, installa i file binari con il comando seguente.

     ```
     choco install kubernetes-helm
     ```

   - Se utilizzi Linux, installa i file binari con i seguenti comandi.

     ```
     curl https://raw.githubusercontent.com/helm/helm/master/scripts/get-helm-3 >
      get_helm.sh
     chmod 700 get_helm.sh
     ./get_helm.sh
     ```

     > **Note**
     >
     > Se ricevi un messaggio che indica la necessità di installare prima `openssl`, è possibile installarlo con il seguente comando.
     >
     > ```
     > sudo yum install openssl
     > ```

2. Per selezionare il nuovo file binario in PATH, chiudere la finestra del terminale corrente e aprirne una nuova.
3. Guarda la versione di Helm che hai installato.

   ```
   helm version | cut -d + -f 1
   ```

   Di seguito viene riportato un output di esempio:

   ```
   v3.9.0
   ```

4. A questo punto, è possibile eseguire qualsiasi comando Helm (ad esempio `helm install` *`chart-name`*) per installare, modificare, eliminare o interrogare i grafici Helm nel cluster. Un nuovo utente di Helm che non ha un grafico specifico da installare, può:

- Sperimentare installando un grafico di esempio. Consulta Install an Example Chart nella Quickstart Guide di Helm.
- Crea un grafico di esempio e invialo ad Amazon ECR. Per ulteriori informazioni, consultare Invio di un grafico Helm nella Guida per l'utente del registro del container di Amazon Elastic.
- Installa un grafico Amazon EKS dal GitHub repository eks-charts o da. ArtifactHub

Assegnazione di tag alle risorse Amazon EKS

È possibile utilizzare tag per aiutarti a gestire le risorse Amazon EKS. In questo argomento viene fornita una panoramica della funzione tag e viene illustrato come creare tag.

Argomenti

> **Note**
>
> I tag sono un tipo di metadati separati dalle etichette e dalle annotazioni di Kubernetes. Per ulteriori informazioni sugli altri tipi di metadati, consulta le sezioni seguenti nella documentazione di Kubernetes:
>
> - Etichette e selettori
> - Annotazioni

Nozioni di base sui tag

Un tag è un'etichetta che si assegna a una AWS risorsa. Ciascun tag è formato da una chiave e da un valore opzionale.

Con i tag, puoi classificare le tue AWS risorse. Ad esempio, puoi categorizzarle in base a scopo, proprietario o ambiente. In presenza di un numero elevato di risorse dello stesso tipo, si possono usare i tag assegnati a una risorsa specifica per identificarla rapidamente. Ad esempio, è possibile definire un set di tag per i cluster Amazon EKS per monitorare il proprietario di ogni cluster e il livello di pila. Ti consigliamo di definire un set coerente di chiavi di tag per ogni tipo di risorsa. È possibile cercare e filtrare le risorse in base ai tag aggiunti.

Dopo aver aggiunto un tag, puoi modificarne le chiavi e i valori oppure rimuovere i tag da una risorsa in qualsiasi momento. Se elimini una risorsa, verranno eliminati anche tutti i tag a essa associati.

I tag non hanno alcun significato semantico per Amazon EKS e vengono interpretati rigorosamente come una stringa di caratteri. È possibile impostare il valore di un tag su una stringa vuota. Tuttavia, non è possibile impostare il valore di un tag su null. Se aggiungi un tag con la stessa chiave di un tag esistente a una risorsa specifica, il nuovo valore sovrascrive quello precedente.

Se utilizzi AWS Identity and Access Management (IAM), puoi controllare quali utenti del tuo AWS account sono autorizzati a gestire i tag.

Tagging delle risorse

I tag sono disponibili per le risorse Amazon EKS seguenti:

- cluster
- gruppi di nodi gestiti
- profili Fargate

Puoi aggiungere un tag a queste risorse utilizzando le opzioni seguenti:

- Se utilizzi la console Amazon EKS, è possibile applicare tag alle risorse nuove o esistenti in qualsiasi momento. A tale scopo, è possibile utilizzare la sheda Tag nella pagina della risorsa interessata. Per ulteriori informazioni, consulta Utilizzo di tag tramite la console.
- Se utilizzi `eksctl`, è possibile applicare i tag alle risorse quando vengono create utilizzando l'opzione `--tags`.
- Se utilizzi l' AWS CLI API Amazon EKS o un AWS SDK, puoi applicare tag a nuove risorse utilizzando il `tags` parametro sull'azione API pertinente. Puoi applicare tag a risorse esistenti utilizzando l'operazione API `TagResource`. Per ulteriori informazioni, consulta TagResource.

Quando utilizzi alcune azioni per la creazione di risorse, puoi anche specificare i tag per la risorsa nello stesso momento in cui la crei. Se i tag non possono essere applicati durante la creazione di una risorsa, la risorsa non potrà essere creata. Mediante questo meccanismo ti assicuri che le risorse a cui desideri applicare tag al momento della creazione vengono create con tag specifici o non vengono create affatto. Se aggiungi tag alle risorse al momento della creazione, non è necessario eseguire script di assegnazione di tag personalizzati dopo la creazione di una risorsa.

I tag non si propagano ad altre risorse associate alla risorsa creata. Ad esempio, i tag del profilo Fargate non vengono propagati ad altre risorse associate al profilo Fargate, come i Pods pianificati con esso.

Limitazioni applicate ai tag

Ai tag si applicano le limitazioni seguenti:

- È possibile associare un massimo di 50 tag a una risorsa.
- Le chiavi dei tag non possono essere ripetute per una risorsa. Ogni chiave di tag deve essere univoca e può avere un solo valore.
- Le chiavi di tag possono contenere fino a 128 caratteri in UTF-8.
- Ogni valore può contenere fino a 256 caratteri UTF-8.
- Se più risorse Servizi AWS e utilizzano lo schema di tagging, limita i tipi di caratteri che usi. Alcuni servizi potrebbero avere restrizioni sui caratteri consentiti. I caratteri generalmente consentiti sono lettere, numeri, spazi e i simboli seguenti: `+ - = . _ : / @`.
- I valori e le chiavi dei tag rispettano la distinzione tra maiuscole e minuscole.
- Non utilizzare `aws:`, `AWS:` o qualsiasi combinazione di maiuscole o minuscole di un tale prefisso per chiavi o valori. Questi sono riservati solo all' AWS uso. Non è possibile modificare né eliminare le chiavi o i valori di tag con tale prefisso. I tag con questo prefisso non rientrano nel tuo tags-per-resource limite.

Tagging delle risorse per la fatturazione

Quando applichi i tag ai cluster Amazon EKS, puoi utilizzarli per l'allocazione dei costi nel tuo Rapporti su costi e utilizzo. I dati di misurazione nel Rapporti su costi e utilizzo mostrano l'utilizzo in tutti i processi dei cluster di Amazon EKS. Per ulteriori informazioni, consulta AWS report su costi e utilizzo nella Guida per l'utente di AWS Billing .

Il tag di allocazione dei costi AWS generato, in particolare`aws:eks:cluster-name`, consente di suddividere i costi delle istanze Amazon EC2 per singolo cluster Amazon EKS in Cost Explorer. Tuttavia, questo tag non rileva le spese del piano di piano di controllo (control-plane). Il tag viene aggiunto automaticamente alle istanze Amazon EC2 che fanno parte di un cluster Amazon EKS. Questo comportamento si verifica indipendentemente dal fatto che le istanze vengano allocate tramite gruppi di nodi gestiti da Amazon EKS, Karpenter o direttamente con Amazon EC2. Questo tag specifico non viene conteggiato ai fini del limite di 50 tag. Per utilizzare il tag, il proprietario dell'account deve attivarlo nella console AWS Billing utilizzando l'API. Quando il proprietario di un account di AWS Organizations gestione attiva il tag, questo viene attivato anche per tutti gli account dei membri dell'organizzazione.

Puoi organizzare le informazioni di fatturazione in base alle risorse con gli stessi valori di chiave di tag. Puoi ad esempio applicare tag a numerose risorse con un nome di applicazione specifico, quindi organizzare le informazioni di fatturazione. In questo modo, puoi visualizzare il costo totale dell'applicazione in più servizi. Per ulteriori informazioni sulla configurazione di un report di allocazione dei costi mediante i tag, consulta Report di allocazione dei costi mensili nella Guida per l'utente di AWS Billing .

> **Note**
>
> Se hai appena abilitato la reportistica, i dati relativi al mese corrente saranno disponibili per la visualizzazione dopo 24 ore.

Cost Explorer è uno strumento di reporting disponibile come parte del piano AWS gratuito. È possibile utilizzare Cost Explorer per visualizzare i grafici delle tue risorse Amazon EKS degli ultimi 13 mesi. Puoi anche prevedere le spese per i prossimi tre mesi. Puoi visualizzare i modelli relativi a quanto spendi in risorse AWS nel tempo. Ad esempio, puoi utilizzarlo per identificare aree che richiedono ulteriore studio e visualizzare le tendenze che puoi utilizzare per comprendere i costi. Puoi anche specificare intervalli di tempo per i dati e visualizzare i dati temporali per mese o per giorno.

Utilizzo di tag tramite la console

Con la console Amazon EKS è possibile gestire i tag associati ai gruppi di nodi gestiti o ai cluster nuovi o esistenti.

Quando selezioni una pagina relativa alle risorse nella console Amazon EKS, viene visualizzato l'elenco delle risorse corrispondenti. Ad esempio, se nel riquadro di navigazione sinistro si seleziona

Cluster, nella console viene visualizzato l'elenco dei cluster Amazon EKS. Quando si seleziona una risorsa in uno di questi elenchi, (ad esempio un cluster specifico), se la risorsa supporta i tag, sarà possibile visualizzare e gestire i tag nella scheda Tag.

Puoi anche utilizzare Tag Editor in AWS Management Console, che fornisce un modo unificato per gestire i tag. Per ulteriori informazioni, consulta Etichettare le AWS risorse con Tag Editor nella Guida per l'utente di AWS Tag Editor.

Aggiunta di tag a una risorsa in fase di creazione

É possibile aggiungere tag ai gruppi di nodi gestiti, ai cluster Amazon EKS ed ai profili Fargate durante la creazione. Per ulteriori informazioni, consulta Creazione di un cluster Amazon EKS.

Aggiunta ed eliminazione di tag in una risorsa

Puoi aggiungere o eliminare i tag associati ai cluster direttamente dalla pagina della risorsa.

Per aggiungere o eliminare un tag su una singola risorsa

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nella barra di navigazione, seleziona l'opzione Regione AWS da usare.
3. Nel pannello di navigazione a sinistra, seleziona Cluster.
4. Scegliere un cluster specifico.
5. Scegliere la scheda Tags (Tag) quindi scegliere Manage tags (Gestisci tag).
6. Nella pagina Gestisci i tag, aggiungi o elimina i tag in base alle esigenze.
   - Per aggiungere un tag, scegli Add tag (Aggiungi tag). Specificare la chiave e il valore per ogni tag.
   - Per eliminare un tag, scegli Rimuovi tag.
7. Ripeti la procedura per ogni tag da aggiungere o eliminare.
8. Scegliere Update (Aggiorna) per terminare.

Utilizzo di tag tramite la CLI, l'API o **`eksctl`**

Utilizza i seguenti AWS CLI comandi o le operazioni API di Amazon EKS per aggiungere, aggiornare, elencare ed eliminare i tag per le tue risorse. È possibile utilizzare solo `eksctl` per aggiungere tag e contemporaneamente creare le nuove risorse con un unico comando.

Supporto dell'assegnazione di tag alle risorse Amazon EKS

Attività	AWS CLI	AWS Tools for Windows PowerShell	Azione API
Aggiungere sovrascrivere uno o più tag.	tag-resource	Add-EKSResourceTag	TagResource
Eliminare uno o più tag.	untag-resource	Remove-EKSResourceTag	UntagResource

I seguenti esempi mostrano come aggiungere o rimuovere tag alle o dalle risorse utilizzando la AWS CLI.

Esempio 1: applicazione di un tag a un cluster esistente

Il comando seguente applica un tag a un cluster esistente.

```
aws eks tag-resource --resource-arn resource_ARN --tags team=devs
```

Esempio 2: rimozione di un tag a un cluster esistente

Il comando seguente elimina un tag da un cluster esistente.

```
aws eks untag-resource --resource-arn resource_ARN --tag-keys tag_key
```

Esempio 3: elencazione dei tag di una risorsa

Il comando seguente elenca i tag associati a una risorsa esistente.

```
aws eks list-tags-for-resource --resource-arn resource_ARN
```

Quando utilizzi alcune azioni per la creazione di risorse, puoi anche specificare i tag per la risorsa nello stesso momento in cui la crei. Le seguenti operazioni supportano l'assegnazione di tag durante la creazione di una risorsa.

Attività	AWS CLI	AWS Tools for Windows PowerShell	Azione API	**eksctl**
Creazione di un cluster	create-cluster	New-EKSCluster	CreateCluster	create cluster
Creazione di un gruppo di nodi gestito*	create-nodegroup	New-EKSNodegroup	CreateNodegroup	create nodegroup
Creazione di un profilo Fargate	create-fargate-profile	New-EKSFargateProfile	CreateFargateProfile.html	create fargateprofile

* Se desideri taggare anche le istanze Amazon EC2 durante la creazione di un gruppo di nodi gestito, crea il gruppo di nodi gestito utilizzando un modello di avvio. Per ulteriori informazioni, consulta Assegnazione di tag a istanze Amazon EC2. Se le istanze esistono già, è possibile taggare manualmente le istanze. Per ulteriori informazioni, consulta Tagging your resources nella Amazon EC2 User Guide.

Service Quotas di Amazon EKS

Amazon EKS si è integrato con Service Quotas, un AWS servizio che puoi utilizzare per visualizzare e gestire le quote da una posizione centrale. Per ulteriori informazioni, consulta Cos'è Service Quotas? nella Guida per l'utente di Service Quotas. Con l'integrazione di Service Quotas, puoi cercare rapidamente il valore delle tue quote di Amazon EKS e dei AWS Fargate servizi utilizzando and. AWS Management Console AWS CLI

AWS Management Console

Per visualizzare le quote dei servizi Amazon EKS e Fargate utilizzando il AWS Management Console

1. Apri la console Service Quotas all'indirizzo https://console.aws.amazon.com/servicequotas/
2. Nel riquadro di navigazione a sinistra, scegliere Servizi AWS.

3. Dall'elenco Servizi AWS, cercare e selezionare Amazon Elastic Kubernetes Service (Amazon EKS) o AWS Fargate.

   Nell'elenco delle quote di servizio, puoi visualizzare il nome della quota di servizio, il valore applicato (se disponibile), la quota AWS predefinita e se il valore della quota è regolabile.
4. Per visualizzare ulteriori informazioni su una quota di servizio, ad esempio la descrizione, scegli il nome della quota.
5. (Facoltativo) Per richiedere un aumento della quota, seleziona la quota che desideri aumentare, seleziona Richiedi un aumento della quota, inserisci o seleziona le informazioni richieste e seleziona Richiedi.

Per utilizzare meglio le quote di servizio utilizzando la AWS Management Console, consulta la Service Quotas User Guide. Per richiedere un aumento delle quote, consultare Richiesta di aumento delle quote nella Guida per l'utente di Service Quotas.

AWS CLI

Per visualizzare le quote dei servizi Amazon EKS e Fargate utilizzando il AWS CLI

Eseguire questo comando per visualizzare le quote Amazon EKS.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code eks \
    --output table
```

Eseguire questo comando per visualizzare le quote di Fargate.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

> **Note**
>
> La quota restituita è il numero di attività Amazon ECS e di Pods Amazon EKS che può eseguire contemporaneamente su Fargate nell'account nella Regione AWS corrente.

Per sfruttare meglio le quote di servizio utilizzando il AWS CLI, consulta la sezione Command *service-quotas*Reference.AWS CLI Per richiedere un aumento delle quote, consultare il comando request-service-quota-increase nella Documentazione di riferimento sui comandi AWS CLI .

Quote del servizio

Nome	Predefinita	Adattabile	Descrizione
Voci di accesso per cluster	Ogni regione supportata: 3.000	No	Il numero massimo di voci di accesso per cluster.
Cluster	Ogni regione supportata: 100	Sì	Il numero massimo di cluster EKS in questo account nella regione corrente.
Gruppi di sicurezza del piano di controllo per cluster	Ogni regione supportata: 4	No	Il numero massimo di gruppi di sicurezza per il piano di controllo (control-plane) del cluster (indicato al momento della creazione del cluster).
Abbonamenti aziendali EKS Anywhere	Ogni regione supportata: 10	Sì	Il numero massimo di abbonamenti aziendali EKS Anywhere in questo account nella regione attuale.
Profili Fargate per cluster	Ogni regione supportata: 10	Sì	Il numero massimo di profili Fargate per cluster.

Nome	Predefinita	Adatta e	Descrizione
Coppie di etichette per selettore di profili Fargate	Ogni regione supportata: 5	Sì	Il numero massimo di coppie di etichette per selettore di profili Fargate.
Gruppi di nodi gestiti per cluster	Ogni regione supportata: 30	Sì	Il numero massimo di gruppi di nodi gestiti per cluster.
Nodi per gruppo di nodi gestiti	Ogni regione supportata: 450	Sì	Il numero massimo di nodi per gruppo di nodi gestiti.
Intervalli CIDR di accesso agli endpoint pubblici per cluster	Ogni regione supportata: 40	No	Il numero massimo di intervalli CIDR di accesso agli endpoint pubblici (specificato al momento della creazione o dell'aggi ornamento del cluster).
Cluster registrati	Ogni regione supportata: 10	Sì	Il numero massimo di cluster registrati in questo account nella regione corrente.
Selettori per profilo Fargate	Ogni regione supportata: 5	Sì	Il numero massimo di selettori per profili Fargate.

Note

I valori predefiniti sono le quote iniziali impostate da. AWS Questi valori predefiniti sono separati dai valori effettivi delle quote applicate e dalle quote massime possibili del servizio. Per ulteriori informazioni, consulta Terminologia di Service Quotas nella Guida per l'utente di Service Quotas.

Queste quote del servizio sono elencate in Amazon Elastic Kubernetes Service (Amazon EKS) nella console Service Quotas. Per chiedere un incremento della quota per i valori indicati come regolabili, consulta Richiesta di incremento di una quota nella Guida per l'utente di Service Quotas.

AWS Fargate quote di servizio

Il servizio AWS Fargate nella console Service Quotas elenca varie quote di servizio. La tabella seguente descrive solo le quote applicabili ad Amazon EKS. È possibile configurare gli allarmi che avvisano quando l'uso si avvicina a una quota di servizio. Per ulteriori informazioni, consulta Creazione di un allarme CloudWatch per monitorare i parametri di utilizzo delle risorse di Fargate.

New Account AWS potrebbe avere quote iniziali inferiori che possono aumentare nel tempo. Fargate monitora costantemente l'utilizzo degli account all'interno di ciascuno di essi Regione AWS, quindi aumenta automaticamente le quote in base all'utilizzo. Puoi chiedere anche un incremento della quota per i valori indicati come regolabili. Per ulteriori informazioni, consulta Richiesta di un aumento di quota nella Guida per l'utente di Service Quotas.

Nome	Predefinita	Adattabile	Descrizione
Conteggio delle risorse vCPU Fargate on demand	6	Sì	Il numero di vCPU Fargate che vengono eseguite simultane amente come Fargate on demand in questo account nella regione corrente.

> **Note**
>
> I valori predefiniti sono le quote iniziali impostate da. AWS Questi valori predefiniti sono separati dai valori effettivi delle quote applicate e dalle quote massime possibili del servizio. Per ulteriori informazioni, consulta Terminologia di Service Quotas nella Guida per l'utente di Service Quotas.

> **Note**
>
> Fargate, inoltre, applica attività Amazon ECS e quote di velocità di avvio dei Pods Amazon EKS. Per ulteriori informazioni, consulta la sezione relativa alla AWS Fargate limitazione delle quote nella guida di Amazon ECS.

Sicurezza in Amazon EKS

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra AWS te e te. Il modello di responsabilità condivisa descrive questo come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi nel AWS cloud. Per Amazon EKS, AWS è responsabile del piano di Kubernetes controllo, che include i nodi del piano di controllo e il `etcd` database. I revisori di terze parti testano e verificano regolarmente l'efficacia della sicurezza come parte dei programmi di conformitàAWS. Per ulteriori informazioni sui programmi di conformità che si applicano ad Amazon EKS, consultare Servizi AWS coperti dal programma di conformità.
- Sicurezza nel cloud - Le seguenti aree sono sotto la responsabilità dell'utente.
  - La configurazione di sicurezza del piano dei dati, compresa la configurazione dei gruppi di sicurezza che autorizzano il passaggio del traffico dal piano di controllo Amazon EKS al VPC del cliente
  - La configurazione dei nodi di lavoro e degli stessi container
  - Il sistema operativo che ospita il nodo di lavoro (inclusi gli aggiornamenti e le patch di sicurezza)
  - Altri software applicativi associati:
    - La configurazione e la gestione dei controlli di rete, come le regole del firewall
    - La gestione di identità a livello di piattaforma e la gestione degli accessi, con o in aggiunta alle IAM
  - Confidenzialità dei dati, requisiti dell'azienda e leggi e normative applicabili

Questa documentazione aiuta a comprendere come applicare il modello di responsabilità condivisa quando si utilizza Amazon EKS. Gli argomenti seguenti descrivono come configurare Amazon EKS per soddisfare gli obiettivi di sicurezza e conformità. Scopri anche come utilizzare altri AWS servizi che ti aiutano a monitorare e proteggere le tue risorse Amazon EKS.

> **Note**
>
> I contenitori Linux sono costituiti da gruppi di controllo (cgroups) e spazi dei nomi che aiutano a limitare ciò a cui un container può accedere, ma tutti i contenitori condividono lo stesso kernel Linux dell'istanza host Amazon EC2. L'esecuzione di un container come utente root (UID 0), o la concessione di un accesso al container a risorse host o spazi dei nomi, come la rete host o lo spazio dei nomi PID host sono fortemente sconsigliati, in quanto ciò riduce l'efficacia dell'isolamento fornito dai container.

Argomenti

Firma dei certificati

L'API Kubernetes Certificates automatizza il provisioning delle credenziali X.509. L'API include un'interfaccia a riga di comando per i client dell'API di Kubernetes per la richiesta e l'acquisizione dei certificati X.509 da un'autorità di certificazione (CA). Puoi utilizzare la risorsa (CSR) `CertificateSigningRequest` per chiedere la firma del certificato da parte di un firmatario designato. Le tue richieste vengono approvate o rifiutate prima che vengano firmate. Kubernetes supporta sia firmatari incorporati che firmatari personalizzati con funzionamenti ben definiti. In questo modo, i client possono prevedere cosa succede alle loro CSR. Per ulteriori informazioni sulla firma dei certificati, consulta firma delle richieste.

Uno dei firmatari integrati è `kubernetes.io/legacy-unknown`. L'API `v1beta1` della risorsa CSR ha onorato questo firmatario "legacy-unknown". Tuttavia, l'API `v1` stabile di CSR non consente l'impostazione di `signerName` su `kubernetes.io/legacy-unknown`.

La versione di Amazon EKS `1.21` e le versioni precedenti consentivano il valore `legacy-unknown` come `signerName` nell'API CSR `v1beta1`. Questa API consente all'autorità di certificazione (CA) di Amazon EKS di generare i certificati. Tuttavia, nella versione `1.22` di Kubernetes, l'API CSR `v1beta1` è stata sostituita dall'API CSR `v1`. che non supporta il signerName di "legacy-unknown". Per utilizzare l'autorità di certificazione (CA) di Amazon EKS per la generazione di certificati sul tuo cluster, devi utilizzare un firmatario personalizzato. come introdotto nella versione `1.22` di Amazon EKS. Per utilizzare la versione `v1` dell'API CSR e generare un nuovo certificato, devi eseguire la migrazione di tutti i manifesti e i client dell'API esistenti. I certificati esistenti creati con l'API `v1beta1` esistente sono validi e funzionanti fino alla scadenza dei certificati. Questo include gli output seguenti:

- Distribuzione di attendibilità: nessuna. Non esiste alcuna distribuzione o attendibilità standard per questo firmatario in un cluster Kubernetes.
- Argomenti consentiti: tutti
- Estensioni x509 consentite: rispetta le estensioni di utilizzo delle chiavi subjectAltName e scarta le altre estensioni
- Utilizzo delle chiavi consentite: non devono includere altri utilizzi oltre ["key encipherment", "digital signature", "server auth"]

> **Note**
>
> La firma dei certificati client non è supportata.

- Scadenza/durata del certificato: 1 anno (predefinita e massima)
- Bit CA consentito/non consentito: non consentito

Generazione CSR di esempio con signerName

Questa procedura mostra come generare un certificato di servizio per il nome DNS `myserver.default.svc` utilizzando `signerName: beta.eks.amazonaws.com/app-serving`. Utilizzala come guida per il tuo ambiente.

1. Esegui il comando `openssl genrsa -out myserver.key 2048` per generare una chiave privata RSA.

```
openssl genrsa -out myserver.key 2048
```

2. Utilizza il comando seguente per generare una richiesta di certificato.

```
openssl req -new -key myserver.key -out myserver.csr -subj "/
CN=myserver.default.svc"
```

3. Genera un valore `base64` per la richiesta CSR e memorizzalo in una variabile da utilizzare in un passaggio successivo.

```
base_64=$(cat myserver.csr | base64 -w 0 | tr -d "\n")
```

4. Per creare un file denominato `mycsr.yaml`, esegui il comando di seguito. Nell'esempio seguente, `beta.eks.amazonaws.com/app-serving` è `signerName`.

```
cat >mycsr.yaml <<EOF
apiVersion: certificates.k8s.io/v1
kind: CertificateSigningRequest
metadata:
  name: myserver
spec:
  request: $base_64
  signerName: beta.eks.amazonaws.com/app-serving
  usages:
    - digital signature
    - key encipherment
    - server auth
EOF
```

5. Invia la CSR.

```
kubectl apply -f mycsr.yaml
```

6. Approva il certificato di servizio.

```
kubectl certificate approve myserver
```

7. Verifica l'emissione del certificato.

```
kubectl get csr myserver
```

Di seguito viene riportato un output di esempio:

```
NAME       AGE     SIGNERNAME                            REQUESTOR
 CONDITION
myserver   3m20s   beta.eks.amazonaws.com/app-serving   kubernetes-admin
 Approved,Issued
```

8. Esporta il certificato emesso.

```
kubectl get csr myserver -o jsonpath='{.status.certificate}'| base64 -d
 > myserver.crt
```

Considerazioni sulla firma dei certificati prima di aggiornare il cluster alla versione 1.24 di Kubernetes

In Kubernetes `1.23` e versioni precedenti, i certificati di servizio `kubelet` con IP non verificabili e nomi alternativi dell'oggetto (SAN) DNS venivano emessi automaticamente con SAN non verificabili. I SAN vengono omessi dal certificato fornito. Nei cluster `1.24` e versioni successive, i certificati di servizio `kubelet` non vengono emessi se non è possibile verificare un SAN. Ciò impedisce il funzionamento dei comandi `kubectl exec` e `kubectl logs`.

Prima di aggiornare il cluster a `1.24`, verifica se nel cluster sono presenti richieste di firma dei certificati (CSR) che non sono state approvate completando i seguenti passaggi:

1. Esegui il comando seguente.

```
kubectl get csr -A
```

Di seguito viene riportato un output di esempio:

```
NAME        AGE   SIGNERNAME                      REQUESTOR
                        REQUESTEDDURATION   CONDITION
csr-7znmf   90m   kubernetes.io/kubelet-serving
 system:node:ip-192-168-42-149.region.compute.internal      <none>
 Approved
csr-9xx5q   90m   kubernetes.io/kubelet-serving
 system:node:ip-192-168-65-38.region.compute.internal      <none>
 Approved, Issued
```

Se l'output restituito mostra una CSR con un firmatario `kubernetes.io/kubelet-serving` che è `Approved` ma non `Issued` per un nodo, devi approvare la richiesta.

2. Approva manualmente la CSR. Sostituire `csr-7znmf` con il proprio valore.

```
kubectl certificate approve csr-7znmf
```

Per approvare automaticamente le CSR in futuro, ti consigliamo di scrivere un controller di approvazione che possa convalidare e approvare automaticamente le CSR contenenti SAN IP o DNS che Amazon EKS non può verificare.

Identity and Access Management per Amazon EKS

AWS Identity and Access Management (IAM) è un programma Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle AWS risorse. Gli amministratori IAM controllano chi è autenticato (accesso effettuato) e autorizzato (dispone di autorizzazioni) a utilizzare risorse Amazon EKS. IAM è un software Servizio AWS che puoi utilizzare senza costi aggiuntivi.

Destinatari

Il modo in cui utilizzi AWS Identity and Access Management (IAM) varia a seconda del lavoro svolto in Amazon EKS.

Utente del servizio:: se si utilizza il servizio Amazon EKS per svolgere il proprio lavoro, l'amministratore fornisce le credenziali e le autorizzazioni necessarie. All'aumentare del numero di caratteristiche Amazon EKS utilizzate per il lavoro, potrebbero essere necessarie ulteriori autorizzazioni. La comprensione della gestione dell'accesso ti consente di richiedere le autorizzazioni corrette all'amministratore. Se non riesci ad accedere a una caratteristica in Amazon EKS, consultare Risoluzione dei problemi di IAM.

Amministratore del servizio: se si è responsabile delle risorse Amazon EKS presso la propria azienda, probabilmente si dispone dell'accesso completo a Amazon EKS. Il compito dell'utente è determinare le caratteristiche e le risorse di Amazon EKS a cui gli utenti del servizio devono accedere. Devi inviare le richieste all'amministratore IAM per cambiare le autorizzazioni degli utenti del servizio. Esamina le informazioni contenute in questa pagina per comprendere i concetti di base relativi a IAM. Per ulteriori informazioni su come la propria azienda può utilizzare IAM con Amazon EKS, consultare Funzionamento di Amazon EKS con IAM.

Amministratore IAM: se si è amministratore IAM, potrebbe essere interessante ottenere informazioni su come scrivere policy per gestire l'accesso ad Amazon EKS. Per visualizzare policy basate su identità Amazon EKS di esempio che possono essere utilizzate in IAM, consultare Esempi di policy basate su identità Amazon EKS.

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. Devi essere autenticato (aver effettuato l' Utente root dell'account AWS accesso AWS) come utente IAM o assumendo un ruolo IAM.

Puoi accedere AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAM Identity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali di Google o Facebook sono esempi di identità federate. Se accedi come identità federata, l'amministratore ha configurato in precedenza la federazione delle identità utilizzando i ruoli IAM. Quando accedi AWS utilizzando la federazione, assumi indirettamente un ruolo.

A seconda del tipo di utente, puoi accedere al AWS Management Console o al portale di AWS accesso. Per ulteriori informazioni sull'accesso a AWS, vedi Come accedere al tuo Account AWS nella Guida per l'Accedi ad AWS utente.

Se accedi a AWS livello di codice, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando (CLI) per firmare crittograficamente le tue richieste utilizzando le tue credenziali. Se non utilizzi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sull'utilizzo del metodo consigliato per firmare autonomamente le richieste, consulta Signing AWS API request nella IAM User Guide.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, ti AWS consiglia di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza del tuo account. Per ulteriori informazioni, consulta Autenticazione a più fattori nella Guida per l'utente di AWS IAM Identity Center e Utilizzo dell'autenticazione a più fattori (MFA) in AWS nella Guida per l'utente di IAM.

Account AWS utente root

Quando si crea un account Account AWS, si inizia con un'identità di accesso che ha accesso completo a tutte Servizi AWS le risorse dell'account. Questa identità è denominata utente Account AWS root ed è accessibile effettuando l'accesso con l'indirizzo e-mail e la password utilizzati per creare l'account. Si consiglia vivamente di non utilizzare l'utente root per le attività quotidiane.

Conserva le credenziali dell'utente root e utilizzarle per eseguire le operazioni che solo l'utente root può eseguire. Per un elenco completo delle attività che richiedono l'accesso come utente root, consulta la sezione Attività che richiedono le credenziali dell'utente root nella Guida per l'utente di IAM.

Utenti e gruppi IAM

Un utente IAM è un'identità interna Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Ove possibile, consigliamo di fare affidamento a credenziali temporanee invece di creare utenti IAM con credenziali a lungo termine come le password e le chiavi di accesso. Tuttavia, per casi d'uso specifici che richiedono credenziali a lungo termine con utenti IAM, si consiglia di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta la pagina Rotazione periodica delle chiavi di accesso per casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente di IAM.

Un gruppo IAM è un'identità che specifica un insieme di utenti IAM. Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, è possibile avere un gruppo denominato Amministratori IAM e concedere a tale gruppo le autorizzazioni per amministrare le risorse IAM.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta Quando creare un utente IAM (invece di un ruolo) nella Guida per l'utente di IAM.

Ruoli IAM

Un ruolo IAM è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche. È simile a un utente IAM, ma non è associato a una persona specifica. Puoi assumere temporaneamente un ruolo IAM in AWS Management Console cambiando ruolo. Puoi assumere un ruolo chiamando un'operazione AWS CLI o AWS API o utilizzando un URL personalizzato. Per ulteriori informazioni sui metodi per l'utilizzo dei ruoli, consulta Utilizzo di ruoli IAM nella Guida per l'utente di IAM.

I ruoli IAM con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene

autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per ulteriori informazioni sulla federazione dei ruoli, consulta Creazione di un ruolo per un provider di identità di terza parte nella Guida per l'utente di IAM. Se utilizzi IAM Identity Center, configura un set di autorizzazioni. IAM Identity Center mette in correlazione il set di autorizzazioni con un ruolo in IAM per controllare a cosa possono accedere le identità dopo l'autenticazione. Per informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .

- Autorizzazioni utente IAM temporanee: un utente IAM o un ruolo può assumere un ruolo IAM per ottenere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso multi-account: è possibile utilizzare un ruolo IAM per permettere a un utente (un principale affidabile) con un account diverso di accedere alle risorse nell'account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per informazioni sulle differenze tra ruoli e policy basate su risorse per l'accesso multi-account, consulta Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente di IAM.
- Accesso a più servizi: alcuni Servizi AWS utilizzano le funzionalità di altri Servizi AWS. Ad esempio, quando effettui una chiamata in un servizio, è comune che tale servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.
  - Sessioni di accesso diretto (FAS): quando utilizzi un utente o un ruolo IAM per eseguire azioni AWS, sei considerato un preside. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra azione in un servizio diverso. FAS utilizza le autorizzazioni del principale che chiama an Servizio AWS, combinate con la richiesta Servizio AWS per effettuare richieste ai servizi downstream. Le richieste FAS vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli delle policy relative alle richieste FAS, consulta la pagina Forward access sessions.
  - Ruolo di servizio: un ruolo di servizio è un ruolo IAM che un servizio assume per eseguire azioni per tuo conto. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione Creazione di un ruolo per delegare le autorizzazioni a un Servizio AWS nella Guida per l'utente di IAM.
  - Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'azione per tuo conto. I ruoli

collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non modificarle.

- Applicazioni in esecuzione su Amazon EC2: puoi utilizzare un ruolo IAM per gestire le credenziali temporanee per le applicazioni in esecuzione su un'istanza EC2 e che AWS CLI effettuano richieste API. AWS Ciò è preferibile all'archiviazione delle chiavi di accesso nell'istanza EC2. Per assegnare un AWS ruolo a un'istanza EC2 e renderlo disponibile per tutte le sue applicazioni, crei un profilo di istanza collegato all'istanza. Un profilo dell'istanza contiene il ruolo e consente ai programmi in esecuzione sull'istanza EC2 di ottenere le credenziali temporanee. Per ulteriori informazioni, consulta Utilizzo di un ruolo IAM per concedere autorizzazioni ad applicazioni in esecuzione su istanze di Amazon EC2 nella Guida per l'utente di IAM.

Per informazioni sull'utilizzo dei ruoli IAM, consulta Quando creare un ruolo IAM (invece di un utente) nella Guida per l'utente di IAM.

Gestione dell'accesso con policy

Puoi controllare l'accesso AWS creando policy e collegandole a AWS identità o risorse. Una policy è un oggetto AWS che, se associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle politiche viene archiviata AWS come documenti JSON. Per ulteriori informazioni sulla struttura e sui contenuti dei documenti delle policy JSON, consulta Panoramica delle policy JSON nella Guida per l'utente di IAM.

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti l'autorizzazione a eseguire azioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM. Successivamente l'amministratore può aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Le policy IAM definiscono le autorizzazioni relative a un'azione, a prescindere dal metodo utilizzato per eseguirla. Ad esempio, supponiamo di disporre di una policy che consente l'azione `iam:GetRole`. Un utente con tale policy può ottenere informazioni sul ruolo dall' AWS Management Console AWS CLI, dall'o dall' AWS API.

Policy basate su identità

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruoli IAM). Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta Creazione di policy IAM nella Guida per l'utente di IAM.

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono integrate direttamente in un singolo utente, gruppo o ruolo. Le politiche gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli nel tuo Account AWS. Le politiche gestite includono politiche AWS gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una policy gestita o una policy inline, consulta Scelta fra policy gestite e policy inline nella Guida per l'utente di IAM.

Policy basate su risorse

Le policy basate su risorse sono documenti di policy JSON che è possibile collegare a una risorsa. Gli esempi più comuni di policy basate su risorse sono le policy di attendibilità dei ruoli IAM e le policy dei bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarle per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non puoi utilizzare le policy AWS gestite di IAM in una policy basata sulle risorse.

Liste di controllo degli accessi (ACL)

Le liste di controllo degli accessi (ACL) controllano quali principali (membri, utenti o ruoli dell'account) hanno le autorizzazioni per accedere a una risorsa. Le ACL sono simili alle policy basate su risorse, sebbene non utilizzino il formato del documento di policy JSON.

Amazon S3 e Amazon VPC sono esempi di servizi che supportano gli ACL. AWS WAF Per maggiori informazioni sulle ACL, consulta Panoramica delle liste di controllo degli accessi (ACL) nella Guida per gli sviluppatori di Amazon Simple Storage Service.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite delle autorizzazioni è una funzione avanzata nella quale si imposta il numero massimo di autorizzazioni che una policy basata su identità può concedere a un'entità IAM (utente o ruolo IAM). È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal` sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità IAM nella Guida per l'utente di IAM.
- Politiche di controllo dei servizi (SCP): le SCP sono politiche JSON che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in. AWS Organizations AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più Account AWS di proprietà dell'azienda. Se abiliti tutte le funzionalità in un'organizzazione, puoi applicare le policy di controllo dei servizi (SCP) a uno o tutti i tuoi account. L'SCP limita le autorizzazioni per le entità negli account dei membri, inclusa ciascuna. Utente root dell'account AWS Per ulteriori informazioni su organizzazioni e policy SCP, consulta la pagina sulle Policy di controllo dei servizi nella Guida per l'utente di AWS Organizations .
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta Policy di sessione nella Guida per l'utente di IAM.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per scoprire come si AWS determina se consentire una richiesta quando sono coinvolti più tipi di policy, consulta Logica di valutazione delle policy nella IAM User Guide.

Funzionamento di Amazon EKS con IAM

Prima di utilizzare IAM per gestire l'accesso ad Amazon EKS, è necessario comprendere quali funzioni IAM sono disponibili per l'uso con Amazon EKS. Per avere una visione di alto livello di come Amazon EKS e altri AWS servizi funzionano con IAM, consulta AWS i servizi che funzionano con IAM nella IAM User Guide.

Argomenti

Policy basate su identità Amazon EKS

Con le policy basate su identità di IAM, è possibile specificare quali azioni e risorse sono consentite o rifiutate, nonché le condizioni in base alle quali le azioni sono consentite o rifiutate. Amazon EKS supporta specifiche operazioni, risorse e chiavi di condizione. Per informazioni su tutti gli elementi utilizzati in una policy JSON, consultare Documentazione di riferimento degli elementi delle policy JSON IAM nella Guida per l'utente di IAM.

Azioni

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire azioni su quali risorse, e in quali condizioni.

L'elemento `Action` di una policy JSON descrive le azioni che è possibile utilizzare per consentire o negare l'accesso a un criterio. Le azioni politiche in genere hanno lo stesso nome dell'operazione AWS API associata. Ci sono alcune eccezioni, ad esempio le azioni di sola autorizzazione che non hanno un'operazione API corrispondente. Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includere le operazioni in una policy per concedere le autorizzazioni per eseguire l'operazione associata.

Le operazioni delle policy in Amazon EKS utilizzano il seguente prefisso prima dell'operazione: `eks:`. Ad esempio, per concedere a qualcuno l'autorizzazione per ottenere le informazioni descrittive su

un cluster Amazon EKS, includere l'operazione `DescribeCluster` nella policy. Le istruzioni della policy devono includere un elemento `Action` o `NotAction`.

Per specificare più operazioni in una sola istruzione, separa ciascuna di esse con una virgola come mostrato di seguito:

```
"Action": ["eks:action1", "eks:action2"]
```

È possibile specificare più azioni tramite caratteri jolly (*). Ad esempio, per specificare tutte le azioni che iniziano con la parola `Describe`, includi la seguente azione:

```
"Action": "eks:Describe*"
```

Per visualizzare un elenco di operazioni di Amazon EKS, consulta Operazioni definite da Amazon Elastic Kubernetes Service in Service Authorization Reference.

Risorse

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento JSON `Resource` della policy specifica l'oggetto o gli oggetti ai quali si applica l'azione. Le istruzioni devono includere un elemento `Resource` o un elemento `NotResource`. Come best practice, specifica una risorsa utilizzando il suo nome della risorsa Amazon (ARN). Puoi eseguire questa operazione per azioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

La risorsa del cluster Amazon EKS dispone del seguente ARN.

```
arn:aws:eks:region-code:account-id:cluster/cluster-name
```

Per ulteriori informazioni sul formato degli ARN, consulta Amazon resource names (ARN) e AWS service namespace.

Ad esempio, per specificare il cluster con il nome *my-cluster* nell'istruzione, utilizzare il seguente ARN:

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/my-cluster"
```

Per specificare tutti i cluster che appartengono a un account specifico e Regione AWS, usa il carattere jolly (*):

```
"Resource": "arn:aws:eks:region-code:111122223333:cluster/*"
```

Alcune operazioni Amazon EKS, ad esempio quelle utili per la creazione di risorse, non possono essere eseguite su una risorsa specifica. In questi casi, è necessario utilizzare il carattere jolly (*).

```
"Resource": "*"
```

Per visualizzare un elenco di tipi di risorse di Amazon EKS, consultare Risorse definite da Amazon Elastic Kubernetes Service in Service Authorization Reference. Per informazioni sulle operazioni con cui è possibile specificare l'ARN di ogni risorsa, consultare Operazioni definite da Amazon Elastic Kubernetes Service.

Chiavi di condizione

Amazon EKS definisce il proprio set di chiavi di condizione e supporta anche l'uso di alcune chiavi di condizione globali. Per visualizzare tutte le chiavi di condizione AWS globali, consulta AWS Global Condition Context Keys nella IAM User Guide.

Puoi impostare le chiavi di condizione quando si associa un provider OpenID Connect al cluster. Per ulteriori informazioni, consulta Policy IAM di esempio.

Tutte le operazioni Amazon EC2 supportano le chiavi di condizione `aws:RequestedRegion` e `ec2:Region`. Per ulteriori informazioni, consulta Esempio: limitazione dell'accesso a uno specifico Regione AWS.

Per un elenco di chiavi di condizione di Amazon EKS, consultare Condizioni per Amazon Elastic Kubernetes Service in Service Authorization Reference. Per informazioni su operazioni e risorse con cui è possibile utilizzare una chiave di condizione, consultare Operazioni definite da Amazon Elastic Kubernetes Service.

Esempi

Per visualizzare esempi di policy basate su identità Amazon EKS, consultare Esempi di policy basate su identità Amazon EKS.

Quando si crea un cluster Amazon EKS, il principale IAM che crea il cluster riceve automaticamente le autorizzazioni `system:masters` nella configurazione del controllo degli accessi basato sul ruolo (RBAC) nel piano di controllo (control-plane) di Amazon EKS. Questo principale IAM non viene visualizzato in una configurazione visibile qualsiasi, quindi assicurati di tenere traccia di quale principale IAM ha originariamente creato il cluster. Per concedere a ulteriori principali IAM la capacità di interagire con il cluster, devi modificare `aws-auth ConfigMap` all'interno di Kubernetes e creare un Kubernetes `rolebinding` o `clusterrolebinding` con il nome di un `group` specificato in `aws-auth ConfigMap`.

Per ulteriori informazioni sull'utilizzo di ConfigMap, vedereConcedi l'accesso alle Kubernetes API .

Policy basate sulle risorse Amazon EKS

Amazon EKS non supporta policy basate su risorse.

Autorizzazione basata su tag Amazon EKS

È possibile allegare i tag alle risorse Amazon EKS o inoltrarli in una richiesta ad Amazon EKS. Per controllare l'accesso basato su tag, fornisci informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* o `aws:TagKeys`. Per ulteriori informazioni sull'assegnazione di tag delle risorse di Amazon EKS, consultare Assegnazione di tag alle risorse Amazon EKS. Per ulteriori informazioni sulle operazioni in cui è possibile utilizzare i tag nelle chiavi di condizione, consulta Operazioni definite da Amazon EKSnel manuale Service Authorization Reference.

Ruoli IAM di Amazon EKS

Un ruolo IAM è un'entità all'interno del tuo AWS account che dispone di autorizzazioni specifiche.

Utilizzo di credenziali temporanee con Amazon EKS

È possibile utilizzare credenziali temporanee per effettuare l'accesso con la federazione, assumere un ruolo IAM o un ruolo multi-account. Ottieni credenziali di sicurezza temporanee chiamando operazioni AWS STS API come AssumeRoleo. GetFederationToken

Amazon EKS supporta l'uso di credenziali temporanee.

Ruoli collegati ai servizi

I ruoli collegati ai AWS servizi consentono ai servizi di accedere alle risorse di altri servizi per completare un'azione per conto dell'utente. I ruoli collegati ai servizi sono visualizzati nell'account IAM e sono di proprietà del servizio. Un amministratore può visualizzare, ma non può modificare le autorizzazioni dei ruoli collegati ai servizi.

Amazon EKS supporta i ruoli collegati ai servizi. Per maggiori dettagli su come creare e gestire i ruoli collegati ai servizi Amazon EKS, consultare Utilizzo di ruoli collegati ai servizi per Amazon EKS.

Ruoli dei servizi

Questa caratteristica consente a un servizio di assumere un ruolo di servizio per conto dell'utente. Questo ruolo consente al servizio di accedere alle risorse in altri servizi per completare un'azione per conto dell'utente. I ruoli dei servizi sono visualizzati nell'account IAM e sono di proprietà dell'account. Ciò significa che un amministratore IAM può modificare le autorizzazioni per questo ruolo. Tuttavia, questo potrebbe pregiudicare la funzionalità del servizio.

Amazon EKS supporta i ruoli del servizio. Per ulteriori informazioni, consulta Ruolo IAM del cluster Amazon EKS e Ruolo IAM del nodo Amazon EKS.

Scelta di un ruolo IAM in Amazon EKS

Quando crei una risorsa cluster in Amazon EKS, devi scegliere un ruolo per consentire ad Amazon EKS di accedere a diverse altre AWS risorse per tuo conto. Se hai già creato un ruolo di servizio in precedenza, Amazon EKS ti fornisce un elenco di ruoli da scegliere. È importante scegliere un ruolo con policy gestite da Amazon EKS allegate. Per ulteriori informazioni, consultare Verifica della presenza di un ruolo del cluster esistente e Verifica della presenza di un ruolo di nodo esistente.

Esempi di policy basate su identità Amazon EKS

Per impostazione predefinita, gli utenti e i ruoli IAM non dispongono dell'autorizzazione per creare o modificare risorse Amazon EKS. Inoltre, non possono eseguire attività utilizzando l' AWS API AWS Management Console AWS CLI, o. Un amministratore IAM deve creare policy IAM che concedono a utenti e ruoli l'autorizzazione per eseguire operazioni API specifiche sulle risorse specificate di cui hanno bisogno. L'amministratore deve quindi allegare queste policy a utenti o IAM che richiedono tali autorizzazioni.

Per informazioni su come creare una policy basata su identità IAM utilizzando questi documenti di policy JSON di esempio, consultare Creazione di policy nella scheda JSON nella Guida per l'utente di IAM.

Quando si crea un cluster Amazon EKS, il principale IAM che crea il cluster riceve automaticamente le autorizzazioni `system:masters` nella configurazione del controllo degli accessi basato sul ruolo (RBAC) nel piano di controllo (control-plane) di Amazon EKS. Questo principale IAM non viene visualizzato in una configurazione visibile qualsiasi, quindi assicurati di tenere traccia di quale principale IAM ha originariamente creato il cluster. Per concedere a ulteriori principali IAM la capacità di interagire con il cluster, devi modificare `aws-auth ConfigMap` all'interno di Kubernetes e creare un Kubernetes `rolebinding` o `clusterrolebinding` con il nome di un `group` specificato in `aws-auth ConfigMap`.

Per ulteriori informazioni sull'utilizzo di ConfigMap, vedereConcedi l'accesso alle Kubernetes API .

Argomenti

- Best practice per le policy
- Utilizzo della console Amazon EKS
- Consentire agli utenti IAM di visualizzare le loro autorizzazioni
- Creazione di un cluster Kubernetes sul Cloud AWS
- Creazione di un cluster Kubernetes locale su un Outpost
- Aggiornamento di un cluster Kubernetes
- Elencare o descrivere tutti i cluster

Best practice per le policy

Le policy basate su identità determinano se qualcuno può creare, accedere o eliminare risorse Amazon EKS nell'account. Queste azioni possono comportare costi aggiuntivi per l' Account AWS. Quando crei o modifichi policy basate su identità, segui queste linee guida e raccomandazioni:

- Inizia con le policy AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le politiche AWS gestite che concedono le autorizzazioni per molti casi d'uso comuni. Sono disponibili nel tuo. Account AWS Ti consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai AWS clienti specifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta Policy gestite da AWS o Policy gestite da AWS per le funzioni dei processi nella Guida per l'utente IAM.
- Applica le autorizzazioni con privilegi minimi: quando imposti le autorizzazioni con le policy IAM, concedi solo le autorizzazioni richieste per eseguire un'attività. Puoi farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come

autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo di IAM per applicare le autorizzazioni, consulta Policy e autorizzazioni in IAM nella Guida per l'utente di IAM.

- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso: per limitare l'accesso ad azioni e risorse puoi aggiungere una condizione alle tue policy. Ad esempio, è possibile scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. Puoi anche utilizzare le condizioni per concedere l'accesso alle azioni del servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per ulteriori informazioni, consulta la sezione Elementi delle policy JSON di IAM: condizione nella Guida per l'utente di IAM.
- Utilizzo di IAM Access Analyzer per convalidare le policy IAM e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano alla sintassi della policy IAM (JSON) e alle best practice di IAM. IAM Access Analyzer offre oltre 100 controlli delle policy e consigli utili per creare policy sicure e funzionali. Per ulteriori informazioni, consulta Convalida delle policy per IAM Access Analyzer nella Guida per l'utente di IAM.
- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede utenti IAM o un utente root nel Account AWS tuo, attiva l'MFA per una maggiore sicurezza. Per richiedere la MFA quando vengono chiamate le operazioni API, aggiungi le condizioni MFA alle policy. Per ulteriori informazioni, consulta Configurazione dell'accesso alle API protetto con MFA nella Guida per l'utente di IAM.

Per maggiori informazioni sulle best practice in IAM, consulta Best practice di sicurezza in IAM nella Guida per l'utente di IAM.

Utilizzo della console Amazon EKS

Per accedere alla console Amazon EKS, un principale IAM deve disporre di un set di autorizzazioni minimo. Queste autorizzazioni consentono al responsabile di elencare e visualizzare i dettagli sulle risorse Amazon EKS nel tuo AWS account. Se crei una policy di basata su identità più restrittiva delle autorizzazioni minime richieste, la console non funzionerà nel modo previsto per i principali associati a tale policy.

Per garantire che i principali IAM possano comunque utilizzare la console Amazon EKS, crea una policy con un nome univoco, ad esempio `AmazonEKSAdminPolicy`. Allega la politica ai principali. Per ulteriori informazioni, consulta Aggiunta e rimozione di autorizzazioni per identità IAM nella Guida per l'utente di IAM.

> **Important**
>
> La policy di esempio riportata di seguito consente a un principale di visualizzare le informazioni sulla scheda Configurazione della console. Per visualizzare le informazioni contenute nelle schede Panoramica e Risorse nella AWS Management Console, il principale ha bisogno inoltre delle autorizzazioni Kubernetes. Per ulteriori informazioni, consulta Autorizzazioni richieste.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": "eks.amazonaws.com"
                }
            }
        }
    ]
}
```

Non è necessario consentire autorizzazioni minime da console per i responsabili che effettuano chiamate solo verso AWS CLI o l'API. AWS Al contrario, è possibile accedere solo alle operazioni che soddisfano l'operazione API che si sta cercando di eseguire.

Consentire agli utenti IAM di visualizzare le loro autorizzazioni

Questo esempio mostra in che modo è possibile creare una policy che consente agli utenti IAM di visualizzare le policy inline e gestite che sono cpllegate alla relativa identità utente. Questa politica

include le autorizzazioni per completare questa azione sulla console o utilizzando l'API o a livello di codice. AWS CLI AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Creazione di un cluster Kubernetes sul Cloud AWS

Regione AWS Puoi sostituirlo Regione AWS con quello in cui desideri creare un cluster Regione AWS . Se viene visualizzato un avviso che indica Le operazioni nella policy non supportano le autorizzazioni a livello di risorsa e richiedono la scelta di **`All resources`** nella AWS Management Console, può essere ignorato senza problemi. Se il tuo account ha già il

*AWSServiceRoleForAmazonEKS*ruolo, puoi rimuovere l'`iam:CreateServiceLinkedRole`azione dalla politica. Se hai mai creato un cluster Amazon EKS nel tuo account, questo ruolo esiste già a meno che sia stato eliminato.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/
eks.amazonaws.com/AWSServiceRoleForAmazonEKS",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "iam:AWSServiceName": "eks"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        }
    ]
}
```

Creazione di un cluster Kubernetes locale su un Outpost

Regione AWS Puoi sostituirlo Regione AWS con quello in cui desideri creare un cluster. Regione AWS Se viene visualizzato un avviso che indica Le operazioni nella policy non supportano le autorizzazioni a livello di risorsa e richiedono la scelta di **`All resources`** nella AWS Management Console, può essere ignorato senza problemi. Se il tuo account dispone già del ruolo `AWSServiceRoleForAmazonEKSLocalOutpost`, puoi rimuovere l'azione `iam:CreateServiceLinkedRole` dalla policy. Se hai già creato un cluster locale Amazon EKS su un Outpost nel tuo account, questo ruolo esiste già a meno che tu non l'abbia eliminato.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:CreateCluster",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        },
        {
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "iam:GetRole"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::111122223333:role/aws-service-role/outposts.eks-
local.amazonaws.com/AWSServiceRoleForAmazonEKSLocalOutpost"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole",
                "iam:ListAttachedRolePolicies"
            ]
            "Resource": "arn:aws:iam::111122223333:role/cluster-role-name"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:TagInstanceProfile",
                "iam:AddRoleToInstanceProfile",
                "iam:GetInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": "arn:aws:iam::*:instance-profile/eks-local-*",
            "Effect": "Allow"
        },
```

```
    ]
}
```

Aggiornamento di un cluster Kubernetes

Regione AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "eks:UpdateClusterVersion",
            "Resource": "arn:aws:eks:us-west-2:111122223333:cluster/my-cluster"
        }
    ]
}
```

Elencare o descrivere tutti i cluster

Questa policy di esempio include le autorizzazioni minime richieste per elencare e descrivere tutti i cluster dell'account. Un principale IAM deve essere in grado di elencare e descrivere i cluster per utilizzare il comando. `update-kubeconfig` AWS CLI

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster",
                "eks:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

Utilizzo di ruoli collegati ai servizi per Amazon EKS

Amazon Elastic Kubernetes Service AWS Identity and Access Management utilizza ruoli collegati ai servizi (IAM). Un ruolo collegato ai servizi è un tipo di ruolo IAM univoco collegato direttamente

ad Amazon EKS. I ruoli collegati ai servizi sono predefiniti da Amazon EKS e includono tutte le autorizzazioni richieste dal servizio per chiamare altri AWS servizi per tuo conto.

Argomenti

Utilizzo dei ruoli per i cluster Amazon EKS

Amazon Elastic Kubernetes Service AWS Identity and Access Management utilizza ruoli collegati ai servizi (IAM). Un ruolo collegato ai servizi è un tipo di ruolo IAM univoco collegato direttamente ad Amazon EKS. I ruoli collegati ai servizi sono predefiniti da Amazon EKS e includono tutte le autorizzazioni richieste dal servizio per chiamare altri AWS servizi per tuo conto.

Un ruolo collegato ai servizi semplifica la configurazione di Amazon EKS perché consente di evitare l'aggiunta manuale delle autorizzazioni necessarie. Amazon EKS definisce le autorizzazioni del ruolo associato ai servizi e, salvo diversamente definito, solo Amazon EKS può assumerne il ruolo. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. Questa procedura protegge le risorse di Amazon EKS perché impedisce la rimozione involontaria delle autorizzazioni di accesso alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consultare Servizi AWS che funzionano con IAM e cercare i servizi che riportano Sì nella colonna Ruolo collegato ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Autorizzazioni del ruolo collegato ai servizi per Amazon EKS

Amazon EKS usa il ruolo collegato ai servizi denominato `AWSServiceRoleForAmazonEKS` – Il ruolo consente ad Amazon EKS di gestire i cluster nell'account. Le policy allegate consentono al ruolo di gestire le seguenti risorse: interfacce di rete, gruppi di sicurezza, registri e VPC.

> **Note**
>
> Il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKS` è distinto dal ruolo richiesto per la creazione del cluster. Per ulteriori informazioni, consulta Ruolo IAM del cluster Amazon EKS.

Ai fini dell'assunzione del ruolo, il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKS` considera attendibili i seguenti servizi:

- `eks.amazonaws.com`

La policy delle autorizzazioni del ruolo consente ad Amazon EKS di eseguire le seguenti operazioni sulle risorse specificate:

- `AmazonEKSServiceRolePolicy`

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato ai servizi per Amazon EKS

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un cluster in AWS Management Console, the o l' AWS API AWS CLI, Amazon EKS crea il ruolo collegato al servizio per te.

Se elimini questo ruolo collegato ai servizi, puoi ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando viene creato un cluster, Amazon EKS crea di nuovo il ruolo collegato ai servizi automaticamente.

Modifica di un ruolo collegato ai servizi per Amazon EKS

Amazon EKS non consente di modificare il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKS`. Dopo aver creato un ruolo collegato al servizio, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato ai servizi per Amazon EKS

Se non è più necessario utilizzare una caratteristica o un servizio che richiede un ruolo collegato ai servizi, ti consigliamo di eliminare quel ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare manualmente.

Pulizia di un ruolo collegato ai servizi

Prima di utilizzare IAM; per eliminare un ruolo collegato al servizio, è necessario prima rimuovere qualsiasi risorsa utilizzata dal ruolo.

> **Note**
>
> Se il servizio Amazon EKS utilizza tale ruolo quando tenti di eliminare le risorse, è possibile che l'eliminazione non riesca. In questo caso, attendi alcuni minuti e quindi ripeti l'operazione.

Per eliminare le risorse Amazon EKS utilizzate dal ruolo **`AWSServiceRoleForAmazonEKS`**.

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra selezionare Clusters (Cluster).
3. Se il cluster dispone di gruppi di nodi o profili Fargate, è necessario eliminarli prima di eliminare il cluster. Per ulteriori informazioni, consulta Eliminazione di un gruppo di nodi gestiti e Eliminazione di un profilo Fargate.
4. Sulla pagina Cluster, scegliere il cluster da eliminare e scegliere Elimina.
5. Digitare il nome del cluster nella finestra di conferma eliminazione, quindi scegliere Elimina.
6. Ripetere questa procedura per tutti gli altri cluster nell'account. Attendere che tutte le operazioni di eliminazione siano completate.

Eliminazione manuale del ruolo collegato ai servizi

Utilizza la console IAM AWS CLI, l'o l' AWS API per eliminare il ruolo collegato al `AWSServiceRoleForAmazonEKS` servizio. Per ulteriori informazioni, consulta Eliminazione del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Regioni supportate per i ruoli collegati ai servizi di Amazon EKS

Amazon EKS supporta l'utilizzo di ruoli collegati ai servizi in tutte le Regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta la sezione Amazon EKS endpoints and quotas (Endpoint e quote di servizio di Amazon EKS).

Utilizzo dei ruoli per i gruppi di nodi Amazon EKS

Amazon EKS utilizza ruoli collegati ai servizi AWS Identity and Access Management (IAM). Un ruolo collegato ai servizi è un tipo di ruolo IAM univoco collegato direttamente ad Amazon EKS. I ruoli collegati ai servizi sono predefiniti da Amazon EKS e includono tutte le autorizzazioni richieste dal servizio per chiamare altri AWS servizi per tuo conto.

Un ruolo collegato ai servizi semplifica la configurazione di Amazon EKS perché consente di evitare l'aggiunta manuale delle autorizzazioni necessarie. Amazon EKS definisce le autorizzazioni del ruolo associato ai servizi e, salvo diversamente definito, solo Amazon EKS può assumerne il ruolo. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. Questa procedura protegge le risorse di Amazon EKS perché impedisce la rimozione involontaria delle autorizzazioni di accesso alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consultare Servizi AWS che funzionano con IAM e cercare i servizi che riportano Sì nella colonna Ruolo collegato ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Autorizzazioni del ruolo collegato ai servizi per Amazon EKS

Amazon EKS usa il ruolo collegato ai servizi denominato `AWSServiceRoleForAmazonEKSNodegroup` – Il ruolo consente ad Amazon EKS di gestire i gruppi di nodi nell'account. Le policy allegate consentono al ruolo di gestire le seguenti risorse: Gruppi di Auto Scaling, gruppi di sicurezza, modelli di avvio e profili dell'istanza IAM.

Ai fini dell'assunzione del ruolo, il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSNodegroup` considera attendibili i seguenti servizi:

- `eks-nodegroup.amazonaws.com`

La policy delle autorizzazioni del ruolo consente ad Amazon EKS di eseguire le seguenti operazioni sulle risorse specificate:

- AWSServiceRoleForAmazonEKSNodegroup

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato ai servizi per Amazon EKS

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando lavori CreateNodegroup nella AWS Management Console, o nell' AWS API AWS CLI, Amazon EKS crea il ruolo collegato al servizio per te.

> **Important**
>
> Questo ruolo collegato al servizio può apparire nell'account, se è stata completata un'operazione in un altro servizio che utilizza le caratteristiche supportate da questo ruolo. Se utilizzavi il servizio Amazon EKS prima del 1° gennaio 2017, quando ha iniziato a supportare ruoli collegati al servizio, Amazon EKS ha creato il AWSServiceRoleForAmazonEKSNodegroup ruolo nel tuo account. Per ulteriori informazioni, consultare Un nuovo ruolo è apparso nel mio account IAM.

Creazione di un ruolo collegato ai servizi in Amazon EKS (API)AWS

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un gruppo di nodi gestiti nella AWS Management Console, o nell' AWS API AWS CLI, Amazon EKS crea il ruolo collegato al servizio per te.

Se elimini questo ruolo collegato ai servizi, puoi ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando si crea un altro gruppo di nodi gestiti, Amazon EKS crea nuovamente il ruolo collegato al servizio.

Modifica di un ruolo collegato ai servizi per Amazon EKS

Amazon EKS non consente di modificare il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSNodegroup`. Dopo aver creato un ruolo collegato al servizio, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia

modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato ai servizi per Amazon EKS

Se non è più necessario utilizzare una caratteristica o un servizio che richiede un ruolo collegato ai servizi, ti consigliamo di eliminare quel ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare manualmente.

Pulizia di un ruolo collegato ai servizi

Prima di utilizzare IAM; per eliminare un ruolo collegato al servizio, è necessario prima rimuovere qualsiasi risorsa utilizzata dal ruolo.

> **Note**
>
> Se il servizio Amazon EKS utilizza tale ruolo quando tenti di eliminare le risorse, è possibile che l'eliminazione non riesca. In questo caso, attendi alcuni minuti e quindi ripeti l'operazione.

Per eliminare le risorse Amazon EKS utilizzate dal ruolo **`AWSServiceRoleForAmazonEKSNodegroup`**.

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, seleziona Cluster.
3. Seleziona la scheda Compute (Calcolo).
4. Nella sezione Gruppi di nodi, scegliere il gruppo di nodi da eliminare.
5. Digitare il nome del gruppo di nodi nella finestra di conferma eliminazione e quindi scegliere Elimina.
6. Ripetere questa procedura per tutti gli altri gruppi di nodi nel cluster. Attendere che tutte le operazioni di eliminazione siano completate.

Eliminazione manuale del ruolo collegato ai servizi

Utilizza la console IAM AWS CLI, l'o l' AWS API per eliminare il ruolo collegato al `AWSServiceRoleForAmazonEKSNodegroup` servizio. Per ulteriori informazioni, consulta Eliminazione del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Regioni supportate per i ruoli collegati ai servizi di Amazon EKS

Amazon EKS supporta l'utilizzo di ruoli collegati ai servizi in tutte le Regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta la sezione Amazon EKS endpoints and quotas (Endpoint e quote di servizio di Amazon EKS).

Utilizzo dei ruoli per i profili Fargate in Amazon EKS

Amazon EKS utilizza ruoli collegati ai servizi AWS Identity and Access Management (IAM). Un ruolo collegato ai servizi è un tipo di ruolo IAM univoco collegato direttamente ad Amazon EKS. I ruoli collegati ai servizi sono predefiniti da Amazon EKS e includono tutte le autorizzazioni richieste dal servizio per chiamare altri AWS servizi per tuo conto.

Un ruolo collegato ai servizi semplifica la configurazione di Amazon EKS perché consente di evitare l'aggiunta manuale delle autorizzazioni necessarie. Amazon EKS definisce le autorizzazioni del ruolo associato ai servizi e, salvo diversamente definito, solo Amazon EKS può assumerne il ruolo. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. Questa procedura protegge le risorse di Amazon EKS perché impedisce la rimozione involontaria delle autorizzazioni di accesso alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consultare Servizi AWS che funzionano con IAM e cercare i servizi che riportano Sì nella colonna Ruolo collegato ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Autorizzazioni del ruolo collegato ai servizi per Amazon EKS

Amazon EKS usa il ruolo collegato ai servizi denominato `AWSServiceRoleForAmazonEKSForFargate`: il ruolo consente ad Amazon EKS Fargate di configurare la rete VPC necessaria per i Pods Fargate. Le policy collegate consentono al ruolo di creare ed eliminare le interfacce di rete elastiche e di descrivere le risorse e le interfacce di rete elastiche.

Ai fini dell'assunzione del ruolo, il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSForFargate` considera attendibili i seguenti servizi:

- `eks-fargate.amazonaws.com`

La policy delle autorizzazioni del ruolo consente ad Amazon EKS di eseguire le seguenti operazioni sulle risorse specificate:

- `AmazonEKSForFargateServiceRolePolicy`

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato ai servizi per Amazon EKS

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un profilo Fargate nell'API AWS Management Console, nella o nell' AWS API AWS CLI, Amazon EKS crea il ruolo collegato al servizio per te.

> **Important**
>
> Questo ruolo collegato al servizio può apparire nell'account, se è stata completata un'operazione in un altro servizio che utilizza le caratteristiche supportate da questo ruolo. Se utilizzavi il servizio Amazon EKS prima del 13 dicembre 2019, quando ha iniziato a supportare ruoli collegati al servizio, Amazon EKS ha creato il AWSServiceRoleForAmazonEKSForFargate ruolo nel tuo account. Per ulteriori informazioni, consulta la sezione A New role appeared in my IAM account (Nel mio account IAM è apparso un nuovo ruolo).

Creazione di un ruolo collegato ai servizi in Amazon EKS (API)AWS

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un profilo Fargate nell'API AWS Management Console, nella o nell' AWS API AWS CLI, Amazon EKS crea il ruolo collegato al servizio per te.

Se elimini questo ruolo collegato ai servizi, puoi ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando si crea un altro gruppo di nodi gestiti, Amazon EKS crea nuovamente il ruolo collegato al servizio.

Modifica di un ruolo collegato ai servizi per Amazon EKS

Amazon EKS non consente di modificare il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSForFargate`. Dopo aver creato un ruolo collegato al servizio,

non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato ai servizi per Amazon EKS

Se non è più necessario utilizzare una caratteristica o un servizio che richiede un ruolo collegato ai servizi, ti consigliamo di eliminare quel ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare manualmente.

Pulizia di un ruolo collegato ai servizi

Prima di utilizzare IAM; per eliminare un ruolo collegato al servizio, è necessario prima rimuovere qualsiasi risorsa utilizzata dal ruolo.

> **Note**
>
> Se il servizio Amazon EKS utilizza tale ruolo quando tenti di eliminare le risorse, è possibile che l'eliminazione non riesca. In questo caso, attendi alcuni minuti e quindi ripeti l'operazione.

Per eliminare le risorse Amazon EKS utilizzate dal ruolo **`AWSServiceRoleForAmazonEKSForFargate`**.

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, seleziona Cluster.
3. Nella pagina Cluster selezionare il cluster.
4. Seleziona la scheda Compute (Calcolo).
5. Se sono presenti profili Fargate nella sezione Fargate profiles (Profili Fargate), selezionare ciascun profilo singolarmente, quindi scegliere Delete (Elimina).
6. Digitare il nome del cluster nella finestra di conferma eliminazione e quindi scegliere Elimina.
7. Ripetere questa procedura per tutti gli altri profili Fargate nel cluster e per ciascun cluster nell'account.

Eliminazione manuale del ruolo collegato ai servizi

Utilizza la console IAM AWS CLI, l'o l' AWS API per eliminare il ruolo collegato al AWSServiceRoleForAmazonEKSForFargate servizio. Per ulteriori informazioni, consulta Eliminazione del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Regioni supportate per i ruoli collegati ai servizi di Amazon EKS

Amazon EKS supporta l'utilizzo di ruoli collegati ai servizi in tutte le Regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta la sezione Amazon EKS endpoints and quotas (Endpoint e quote di servizio di Amazon EKS).

Utilizzo di ruoli per connettere un cluster Kubernetes ad Amazon EKS

Amazon EKS utilizza ruoli collegati ai servizi AWS Identity and Access Management (IAM). Un ruolo collegato ai servizi è un tipo di ruolo IAM univoco collegato direttamente ad Amazon EKS. I ruoli collegati ai servizi sono predefiniti da Amazon EKS e includono tutte le autorizzazioni richieste dal servizio per chiamare altri AWS servizi per tuo conto.

Un ruolo collegato ai servizi semplifica la configurazione di Amazon EKS perché consente di evitare l'aggiunta manuale delle autorizzazioni necessarie. Amazon EKS definisce le autorizzazioni del ruolo associato ai servizi e, salvo diversamente definito, solo Amazon EKS può assumerne il ruolo. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. Questa procedura protegge le risorse di Amazon EKS perché impedisce la rimozione involontaria delle autorizzazioni di accesso alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consultare Servizi AWS che funzionano con IAM e cercare i servizi che riportano Sì nella colonna Ruolo collegato ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Autorizzazioni del ruolo collegato ai servizi per Amazon EKS

Amazon EKS usa il ruolo collegato ai servizi denominato`AWSServiceRoleForAmazonEKSConnector`: il ruolo consente ad Amazon EKS di connettere cluster Kubernetes. Le policy allegate consentono al ruolo di gestire le risorse necessarie per connettersi al cluster Kubernetes registrato.

Ai fini dell'assunzione del ruolo, il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSConnector` considera attendibili i seguenti servizi:

- `eks-connector.amazonaws.com`

La policy delle autorizzazioni del ruolo consente ad Amazon EKS di eseguire le seguenti operazioni sulle risorse specificate:

- `AmazonEKSConnectorServiceRolePolicy`

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato ai servizi per Amazon EKS

Non devi creare manualmente un ruolo collegato ai servizi per collegare un cluster. Quando connetti un cluster nell'API AWS Management Console AWS CLI, nell'o nell' AWS API`eksctl`, Amazon EKS crea il ruolo collegato al servizio per te.

Se elimini questo ruolo collegato ai servizi, puoi ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando viene collegato un cluster, Amazon EKS crea di nuovo il ruolo collegato ai servizi automaticamente.

Modifica di un ruolo collegato ai servizi per Amazon EKS

Amazon EKS non consente di modificare il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSConnector`. Dopo aver creato un ruolo collegato al servizio, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato ai servizi per Amazon EKS

Se non è più necessario utilizzare una caratteristica o un servizio che richiede un ruolo collegato ai servizi, ti consigliamo di eliminare quel ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare manualmente.

Pulizia di un ruolo collegato ai servizi

Prima di utilizzare IAM; per eliminare un ruolo collegato al servizio, è necessario prima rimuovere qualsiasi risorsa utilizzata dal ruolo.

> **Note**
>
> Se il servizio Amazon EKS utilizza tale ruolo quando tenti di eliminare le risorse, è possibile che l'eliminazione non riesca. In questo caso, attendi alcuni minuti e quindi ripeti l'operazione.

Per eliminare le risorse Amazon EKS utilizzate dal ruolo **`AWSServiceRoleForAmazonEKSConnector`**.

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, seleziona Cluster.
3. Nella pagina Cluster selezionare il cluster.
4. Selezionare la scheda Annulla registrazione e quindi selezionare la scheda Ok.

Eliminazione manuale del ruolo collegato ai servizi

Utilizza la console IAM AWS CLI, l'o l' AWS API per eliminare il ruolo collegato al AWSServiceRoleForAmazonEKSConnector servizio. Per ulteriori informazioni, consultare Eliminazione del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Utilizzo dei ruoli per i cluster locali Amazon EKS su Outpost

Amazon Elastic Kubernetes Service AWS Identity and Access Management utilizza ruoli collegati ai servizi (IAM). Un ruolo collegato ai servizi è un tipo di ruolo IAM univoco collegato direttamente ad Amazon EKS. I ruoli collegati ai servizi sono predefiniti da Amazon EKS e includono tutte le autorizzazioni richieste dal servizio per chiamare altri AWS servizi per tuo conto.

Un ruolo collegato ai servizi semplifica la configurazione di Amazon EKS perché consente di evitare l'aggiunta manuale delle autorizzazioni necessarie. Amazon EKS definisce le autorizzazioni del ruolo associato ai servizi e, salvo diversamente definito, solo Amazon EKS può assumerne il ruolo. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. Questa procedura protegge le risorse di Amazon EKS perché impedisce la rimozione involontaria delle autorizzazioni di accesso alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consultare Servizi AWS che funzionano con IAM e cercare i servizi che riportano Sì nella colonna Ruolo collegato ai servizi. Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato ai servizi per tale servizio.

Autorizzazioni del ruolo collegato ai servizi per Amazon EKS

Amazon EKS utilizza il ruolo collegato ai servizi denominato `AWSServiceRoleForAmazonEKSLocalOutpost`: il ruolo consente ad Amazon EKS di gestire i cluster locali nell'account. Le policy collegate consentono al ruolo di gestire le seguenti risorse: interfacce di rete, gruppi di sicurezza, registri e istanze Amazon EC2.

> **Note**
>
> Il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSLocalOutpost` è distinto dal ruolo richiesto per la creazione del cluster. Per ulteriori informazioni, consulta Ruolo IAM del cluster Amazon EKS.

Ai fini dell'assunzione del ruolo, il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSLocalOutpost` considera attendibili i seguenti servizi:

- `outposts.eks-local.amazonaws.com`

La policy delle autorizzazioni del ruolo consente ad Amazon EKS di eseguire le seguenti operazioni sulle risorse specificate:

- `AmazonEKSServiceRolePolicy`

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato ai servizi per Amazon EKS

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un cluster in AWS Management Console, the o l' AWS API AWS CLI, Amazon EKS crea il ruolo collegato al servizio per te.

Se elimini questo ruolo collegato ai servizi, puoi ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando viene creato un cluster, Amazon EKS crea di nuovo il ruolo collegato ai servizi automaticamente.

Modifica di un ruolo collegato ai servizi per Amazon EKS

Amazon EKS non consente di modificare il ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSLocalOutpost`. Dopo aver creato un ruolo collegato al servizio, non puoi modificarne il nome, perché potrebbero farvi riferimento diverse entità. Puoi tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato ai servizi per Amazon EKS

Se non è più necessario utilizzare una caratteristica o un servizio che richiede un ruolo collegato ai servizi, ti consigliamo di eliminare quel ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente. Tuttavia, è necessario effettuare la pulizia delle risorse associate al ruolo collegato ai servizi prima di poterlo eliminare manualmente.

Pulizia di un ruolo collegato ai servizi

Prima di utilizzare IAM; per eliminare un ruolo collegato al servizio, è necessario prima rimuovere qualsiasi risorsa utilizzata dal ruolo.

> **Note**
>
> Se il servizio Amazon EKS utilizza tale ruolo quando tenti di eliminare le risorse, è possibile che l'eliminazione non riesca. In questo caso, attendi alcuni minuti e quindi ripeti l'operazione.

Per eliminare le risorse Amazon EKS utilizzate dal ruolo **`AWSServiceRoleForAmazonEKSLocalOutpost`**.

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nel pannello di navigazione a sinistra, scegli Clusters (Cluster) Amazon EKS.

3. Se il cluster dispone di gruppi di nodi o profili Fargate, è necessario eliminarli prima di eliminare il cluster. Per ulteriori informazioni, consulta Eliminazione di un gruppo di nodi gestiti e Eliminazione di un profilo Fargate.
4. Sulla pagina Cluster, scegliere il cluster da eliminare e scegliere Elimina.
5. Digitare il nome del cluster nella finestra di conferma eliminazione, quindi scegliere Elimina.
6. Ripetere questa procedura per tutti gli altri cluster nell'account. Attendere che tutte le operazioni di eliminazione siano completate.

Eliminazione manuale del ruolo collegato ai servizi

Utilizza la console IAM AWS CLI, l'o l' AWS API per eliminare il ruolo collegato al `AWSServiceRoleForAmazonEKSLocalOutpost` servizio. Per ulteriori informazioni, consulta Eliminazione del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Regioni supportate per i ruoli collegati ai servizi di Amazon EKS

Amazon EKS supporta l'utilizzo di ruoli collegati ai servizi in tutte le Regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta la sezione Amazon EKS endpoints and quotas (Endpoint e quote di servizio di Amazon EKS).

Ruolo IAM del cluster Amazon EKS

Il ruolo IAM del cluster Amazon EKS è richiesto per ciascun cluster. I cluster Kubernetes gestiti da Amazon EKS utilizzano questo ruolo per gestire i nodi e il Provider cloud legacy utilizza questo ruolo per creare sistemi di bilanciamento del carico con Elastic Load Balancing per i servizi.

Prima di poter creare cluster Amazon EKS, devi creare un ruolo IAM con una delle seguenti policy IAM:

- `AmazonEKSClusterPolicy`
- Una policy IAM personalizzata. Le autorizzazioni minime che seguono consentono al cluster Kubernetes di gestire i nodi, ma non consentono al Provider cloud legacy di creare sistemi di bilanciamento del carico con Elastic Load Balancing. La tua policy IAM personalizzata deve disporre almeno delle seguenti autorizzazioni:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
```

```
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:*:*:instance/*",
      "Condition": {
        "ForAnyValue:StringLike": {
          "aws:TagKeys": "kubernetes.io/cluster/*"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeVpcs",
        "ec2:DescribeDhcpOptions",
        "ec2:DescribeAvailabilityZones",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
  ]
}
```

> **Note**
>
> Prima del 3 ottobre 2023, ClusterPolicyAmazonEks aveva bisogno del ruolo IAM per ogni cluster.
> Prima del 16 aprile 2020, AmazonEks e ServicePolicy ClusterPolicy AmazonEks erano obbligatori e il nome suggerito per il ruolo era. `eksServiceRole` Con il ruolo AWSServiceRoleForAmazonEKS collegato ai servizi, la ServicePolicy policy AmazonEks non è più richiesta per i cluster creati a partire dal 16 aprile 2020.

Verifica della presenza di un ruolo del cluster esistente

Per verificare se l'account dispone già di un ruolo del cluster Amazon EKS, utilizzare la procedura indicata di seguito.

Per verificare la presenza di **`eksClusterRole`** nella console IAM

1. Aprire la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Cerca l'elenco dei ruoli per `eksClusterRole`. Se un ruolo che include `eksClusterRole` non esiste, consulta Creazione del ruolo del cluster Amazon EKS per crearlo. Se un ruolo che include `eksClusterRole` esiste, seleziona il ruolo per visualizzare le policy allegate.
4. Selezionare Autorizzazioni.
5. Assicurati che la policy gestita da ClusterPolicy AmazonEks sia associata al ruolo. Se la policy è allegata, il ruolo del cluster Amazon EKS è configurato correttamente.
6. Scegli Trust relationships (Relazioni di attendibilità), quindi scegli Edit trust policy (Modifica policy di attendibilità).
7. Verifica che la relazione di trust includa la policy seguente. Se la relazione di attendibilità corrisponde alla policy seguente, scegli Cancel (Annulla). Se la relazione di attendibilità non corrisponde, copia la policy nella finestra Modifica policy di attendibilità e scegli Aggiorna policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Creazione del ruolo del cluster Amazon EKS

Puoi usare AWS Management Console o the AWS CLI per creare il ruolo del cluster.

AWS Management Console

Creazione del ruolo del cluster Amazon EKS nella console IAM

1. Aprire la console IAM all'indirizzo https://console.aws.amazon.com/iam/.

2. Scegli Ruoli, quindi Crea ruolo.
3. In Trusted entity type (Tipo di entità attendibile), scegli AWS service (Servizio).
4. Nell'elenco a discesa Casi d'uso per altri Servizi AWS, scegli EKS.
5. Scegli EKS - Cluster per il tuo caso d'uso, quindi scegli Next (Successivo).
6. Nella scheda Aggiungi autorizzazioni, scegli Successivo.
7. Per Nome ruolo, inserisci un nome univoco per il ruolo, ad esempio **`eksClusterRole`**.
8. Per Description (Descrizione), inserisci un testo descrittivo come **`Amazon EKS - Cluster role`**.
9. Scegli Crea ruolo.

AWS CLI

1. Copiare i seguenti contenuti in un file denominato *`cluster-trust-policy.json`*.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "eks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Crea il ruolo. Puoi sostituire ***`eksClusterRole`*** con un nome a tua scelta.

```
aws iam create-role \
  --role-name eksClusterRole \
  --assume-role-policy-document file://"cluster-trust-policy.json"
```

3. Allega la policy IAM richiesta al ruolo.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSClusterPolicy \
  --role-name eksClusterRole
```

Ruolo IAM del nodo Amazon EKS

Il `kubelet` daemon del nodo Amazon EKS effettua chiamate alle AWS API per tuo conto. I nodi ricevono le autorizzazioni per queste chiamate API attraverso un profilo dell'istanza IAM e le policy associate. Prima di avviare i nodi e registrarli in un cluster, devi creare un ruolo IAM che i nodi possano utilizzare all'avvio. Questo requisito si applica ai nodi di lavoro avviati con l'AMI ottimizzata per Amazon EKS fornita da Amazon o con altre AMI di nodi che intendi utilizzare. Inoltre, questo requisito si applica sia ai gruppi di nodi gestiti sia ai nodi autogestiti.

> **Note**
>
> Non è possibile utilizzare lo stesso ruolo utilizzato per creare i cluster.

Prima di creare i nodi, è necessario creare un ruolo IAM con le seguenti autorizzazioni:

- Autorizzazioni per consentire a `kubelet` di descrivere le risorse Amazon EC2 nel VPC, come quelle fornite dalla policy `AmazonEKSWorkerNodePolicy`. Questa policy fornisce anche le autorizzazioni per il Pod Identity Agent di Amazon EKS.
- Autorizzazioni per consentire a `kubelet` di utilizzare le immagini dei container da Amazon Elastic Container Registry (Amazon ECR), come previsto dalla policy `AmazonEC2ContainerRegistryReadOnly`. Le autorizzazioni per utilizzare le immagini dei container da Amazon Elastic Container Registry (Amazon ECR) sono necessarie perché i componenti aggiuntivi integrati per le reti eseguono pod che utilizzano immagini di container provenienti da Amazon ECR.
- (Facoltativo) Autorizzazioni per consentire al Pod Identity Agent di Amazon EKS di utilizzare l'azione `eks-auth:AssumeRoleForPodIdentity` per recuperare le credenziali per i pod. Se non utilizzi WorkerNodePolicyAmazonEks, devi fornire questa autorizzazione oltre alle autorizzazioni EC2 per utilizzare EKS Pod Identity.
- (Facoltativo) Se non si usa Pod Identity di EKS o IRSA per fornire le autorizzazioni ai pod VPC CNI, è necessario fornire le autorizzazioni per il VPC CNI sul ruolo dell'istanza. È possibile utilizzare la policy gestita `AmazonEKS_CNI_Policy` (se il cluster è stato creato con la famiglia `IPv4`) o una policy IPv6 creata (se il cluster è stato creato con la famiglia `IPv6`). Invece di allegare la policy a questo ruolo, tuttavia, consigliamo di allegarla a un ruolo separato utilizzato specificamente per il componente aggiuntivo CNI di Amazon VPC. Per ulteriori informazioni sulla creazione di un ruolo separato per il componente aggiuntivo CNI di Amazon VPC, consulta

Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

> **Note**
>
> Prima del 3 ottobre 2023, `AmazonEKSWorkerNodePolicy` e `AmazonEC2ContainerRegistryReadOnly` erano obbligatori nel ruolo IAM per ciascun gruppo di nodi gestiti.
> I gruppi di nodi Amazon EC2 devono avere un ruolo IAM diverso dal profilo Fargate. Per ulteriori informazioni, consulta Ruolo IAM per l'esecuzione del Pod Amazon EKS.

Verifica della presenza di un ruolo di nodo esistente

Per controllare se l'account dispone già di un ruolo di nodo Amazon EKS, utilizzare la procedura indicata di seguito.

Per verificare la presenza di **`eksNodeRole`** nella console IAM

1. Aprire la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Ricerca `eksNodeRole`, `AmazonEKSNodeRole` o `NodeInstanceRole` nell'elenco di ruoli. Se un ruolo con uno di questi nomi non esiste, consulta Creazione del ruolo IAM del nodo Amazon EKS per creare il ruolo. Se esiste un ruolo che contiene `eksNodeRole`, `AmazonEKSNodeRole` o `NodeInstanceRole`, seleziona il ruolo per visualizzare le policy allegate.
4. Seleziona Autorizzazioni.
5. Assicurati che le policy gestite di WorkerNodePolicy AmazonEks e ContainerRegistryReadOnly AmazoneEC2 siano collegate al ruolo o che sia allegata una policy personalizzata con le autorizzazioni minime.

   > **Note**
   >
   > Se al ruolo è invece collegata la policy AmazonEKS_CNI_Policy, è consigliabile rimuoverla e allegarla a un ruolo IAM mappato all'account di servizio Kubernetes `aws-node`. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

6. Scegli Trust relationships (Relazioni di attendibilità), quindi scegli Edit trust policy (Modifica policy di attendibilità).
7. Verifica che la relazione di trust includa la policy seguente. Se la relazione di attendibilità corrisponde alla policy seguente, scegli Cancel (Annulla). Se la relazione di attendibilità non corrisponde, copia la policy nella finestra Modifica policy di attendibilità e scegli Aggiorna policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Creazione del ruolo IAM del nodo Amazon EKS

Puoi creare il ruolo IAM del nodo con o. AWS Management Console AWS CLI

AWS Management Console

Creazione del ruolo del nodo Amazon EKS nella console IAM

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Nella pagina Ruoli, seleziona Crea ruolo.
4. Nella pagina Seleziona un'entità attendibile, esegui le operazioni seguenti:
   a. Nella sezione Tipo di entità attendibile, scegli Servizio AWS .
   b. Per Use case (Caso d'uso), seleziona EC2.
   c. Seleziona Successivo.
5. Nella pagina Aggiungi autorizzazioni, collega una policy personalizzata o esegui le operazioni seguenti:

a. Nella casella Filtra policy, inserisci **`AmazonEKSWorkerNodePolicy`**.

b. Seleziona la casella di controllo a sinistra di WorkerNodePolicyAmazonEks nei risultati della ricerca.

c. Scegli Cancella filtri.

d. Nella casella Filtra policy, inserisci **`AmazonEC2ContainerRegistryReadOnly`**.

e. Seleziona la casella di controllo a sinistra di AmazoneC2 ContainerRegistryReadOnly nei risultati della ricerca.

Inoltre, a questo ruolo o a un ruolo diverso mappato all'account del servizio Kubernetes `aws-node` deve essere collegata la policy gestita AmazonEKS_CNI_Policy o una policy IPv6 creata da te. Si consiglia di assegnare la policy al ruolo associato all'account del servizio Kubernetes anziché assegnarlo a questo ruolo. Per ulteriori informazioni, consulta la pagina Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

f. Seleziona Next (Successivo).

6. Nella pagina Name, review, and create (Assegna un nome, rivedi e crea), esegui le operazioni seguenti:

a. Per Role name (Nome ruolo), inserisci un nome univoco per il ruolo, ad esempio **`AmazonEKSNodeRole`**.

b. In Descrizione, sostituisci il testo corrente con un testo descrittivo come **`Amazon EKS - Node role`**.

c. In Aggiungi tag (facoltativo), aggiungi metadati al ruolo collegando i tag come coppie chiave-valore. Per ulteriori informazioni sull'utilizzo di tag in IAM, consulta la sezione Applicazione di tag alle risorse IAM nella Guida per l'utente di IAM.

d. Scegli Crea ruolo.

AWS CLI

1. Per creare il file `node-role-trust-relationship.json`, emetti il seguente comando:

```
cat >node-role-trust-relationship.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

2. Crea il ruolo IAM.

```
aws iam create-role \
  --role-name AmazonEKSNodeRole \
  --assume-role-policy-document file://"node-role-trust-relationship.json"
```

3. Allegare al ruolo IAM le due policy gestite IAM richieste.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSWorkerNodePolicy \
  --role-name AmazonEKSNodeRole
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEC2ContainerRegistryReadOnly \
  --role-name AmazonEKSNodeRole
```

4. Allega una delle seguenti policy IAM al ruolo IAM a seconda della famiglia di IP con cui hai creato il cluster. La policy deve essere allegata a questo ruolo o a un ruolo associato all'account di servizio `aws-node` Kubernetesche viene utilizzato per il Amazon VPC CNI plugin for Kubernetes. Si consiglia di assegnare la policy al ruolo associato all'account del servizio Kubernetes. Per assegnare la policy al ruolo associato all'account del servizio Kubernetes, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

   - IPv4

     ```
     aws iam attach-role-policy \
       --policy-arn arn:aws:iam::aws:policy/AmazonEKS_CNI_Policy \
       --role-name AmazonEKSNodeRole
     ```

   - IPv6

     1. Copia il testo seguente e salvalo in un file denominato *vpc-cni-ipv6-policy*.json.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssignIpv6Addresses",
                "ec2:DescribeInstances",
                "ec2:DescribeTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:network-interface/*"
            ]
        }
    ]
}
```

2. Creare la policy IAM.

```
aws iam create-policy --policy-name AmazonEKS_CNI_IPv6_Policy --policy-
document file://vpc-cni-ipv6-policy.json
```

3. Allega la policy IAM al ruolo IAM. Sostituisci *111122223333* con l'ID del tuo account.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::111122223333:policy/AmazonEKS_CNI_IPv6_Policy \
  --role-name AmazonEKSNodeRole
```

Ruolo IAM per l'esecuzione del Pod Amazon EKS

Il ruolo di Pod esecuzione di Amazon EKS è necessario per l'esecuzione Pods sull' AWS Fargate infrastruttura.

Quando il cluster crea un'Pods AWS Fargate infrastruttura, i componenti in esecuzione sull'infrastruttura Fargate devono effettuare chiamate alle AWS API per conto dell'utente. In questo modo possono eseguire azioni come estrarre le immagini dei container da Amazon ECR o indirizzare i log ad altri AWS servizi. Il ruolo di esecuzione del Pod Amazon EKS fornisce le autorizzazioni IAM per eseguire questa operazione.

Quando crei un profilo Fargate, devi specificare un ruolo di esecuzione del Pod per i componenti Amazon EKS che vengono eseguiti su infrastruttura Fargate utilizzando il profilo. Questo ruolo viene aggiunto al controllo degli accessi basato sul ruolo (RBAC) di Kubernetes del cluster per l'autorizzazione. Ciò consente al `kubelet` in esecuzione sull'infrastruttura Fargate di registrarsi con il cluster Amazon EKS in modo che possa essere visualizzato nel cluster come nodo.

> **Note**
>
> Il profilo Fargate deve avere un ruolo IAM diverso dai gruppi di nodi Amazon EC2.

> **Important**
>
> I container in esecuzione nel Pod Fargate non possono assumere le autorizzazioni IAM associate a un ruolo di esecuzione del Pod. Per concedere ai contenitori del tuo Fargate Pod le autorizzazioni per accedere ad altri AWS servizi, devi usare. Ruoli IAM per gli account di servizio

Prima di creare un profilo Fargate, è necessario creare un ruolo IAM con la `AmazonEKSFargatePodExecutionRolePolicy`.

Verifica della presenza di un ruolo di esecuzione del Pod esistente configurato correttamente

Per verificare se l'account dispone già di un ruolo di esecuzione del Pod Amazon EKS configurato correttamente, utilizza la procedura indicata di seguito. Per prevenire il problema di sicurezza "confused deputy", è importante che il ruolo limiti l'accesso in base a `SourceArn`. È possibile

modificare il ruolo di esecuzione secondo necessità per includere il supporto per i profili Fargate su altri cluster.

Verifica della presenza di un ruolo di esecuzione del Pod Amazon EKS nella console IAM

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Nella pagina Ruoli, cerca l'elenco dei ruoli per AmazonEks FargatePodExecutionRole. Se il ruolo non esiste, consulta Creazione del ruolo di esecuzione del Pod Amazon EKS per crearlo. Se il ruolo è presente, selezionalo.
4. Nella pagina FargatePodExecutionRoleAmazonEks, procedi come segue:
   a. Seleziona Autorizzazioni.
   b. Assicurati che la policy gestita da FargatePodExecutionRolePolicyAmazonEks di Amazon sia associata al ruolo.
   c. Scegli Trust relationships (Relazioni di trust).
   d. Seleziona Edit trust policy (Modifica policy di attendibilità).
5. Nella pagina Edit trust policy (Modifica policy di attendibilità), verifica che la relazione di attendibilità contenga la policy seguente e una riga per i profili Fargate nel cluster. In tal caso, scegli Cancel (Annulla).

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Se la policy corrisponde ma non contiene una riga che specifica i profili Fargate sul cluster, aggiungi la riga riportata di seguito nella parte superiore dell'oggetto `ArnLike`. Sostituisci *`region-code`* con la Regione AWS in cui si trova il cluster, *`111122223333`* con il tuo ID account e *`my-cluster`* con il nome del cluster.

```
"aws:SourceArn": "arn:aws:eks:region-code:111122223333:fargateprofile/my-cluster/
*",
```

Se la policy non corrisponde, copia la policy precedente nel modulo e scegli Update policy (Aggiorna policy). Sostituisci *`region-code`* con la Regione AWS in cui si trova il cluster. * Sostituisci *`111122223333`* con il tuo ID account e *`my-cluster`* con il nome del cluster. Se vuoi usare lo stesso ruolo per tutti i cluster dell'account, sostituisci *`my-cluster`* con *.

Creazione del ruolo di esecuzione del Pod Amazon EKS

Se non disponi già del ruolo di Pod esecuzione di Amazon EKS per il tuo cluster, puoi utilizzare AWS Management Console o the AWS CLI per crearlo.

AWS Management Console

Creazione di un ruolo di esecuzione PodAWS Fargate tramite la AWS Management Console

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Nella pagina Ruoli, seleziona Crea ruolo.
4. Nella pagina Seleziona un'entità attendibile, esegui le operazioni seguenti:
   a. Nella sezione Trusted entity type (Tipo di entità attendibile), scegli AWS service (Servizio).
   b. Nell'elenco a discesa Casi d'uso per altri Servizi AWS, scegli EKS.
   c. Scegli Pod EKS - Fargate.
   d. Scegli Next (Successivo).
5. Nella pagina Add permissions (Aggiungi autorizzazioni), scegli Next (Successivo).
6. Nella pagina Name, review, and create (Assegna un nome, rivedi e crea), esegui le operazioni seguenti:

a. Per Role name (Nome ruolo), inserisci un nome univoco per il ruolo, ad esempio **`AmazonEKSFargatePodExecutionRole`**.

b. In Aggiungi tag (facoltativo), aggiungi metadati al ruolo collegando i tag come coppie chiave-valore. Per ulteriori informazioni sull'utilizzo di tag in IAM, consulta la sezione Applicazione di tag alle risorse IAM nella Guida per l'utente di IAM.

c. Scegli Crea ruolo.

7. Nella pagina Ruoli, cerca l'elenco dei ruoli per AmazonEks FargatePodExecutionRole. Seleziona il ruolo.

8. Nella pagina FargatePodExecutionRoleAmazonEks, procedi come segue:

a. Scegli Trust relationships (Relazioni di trust).

b. Seleziona Edit trust policy (Modifica policy di attendibilità).

9. Nella pagina Modifica policy di attendibilità, effettua le operazioni seguenti:

a. Copia e incolla il contenuto seguente nel modello Modifica policy di attendibilità. Sostituisci il *`codice regionale us-iso-east`* con Regione AWS quello in cui si trova il tuo cluster. * Sostituisci *`111122223333`* con il tuo ID account e *`my-cluster`* con il nome del cluster. Se vuoi usare lo stesso ruolo per tutti i cluster dell'account, sostituisci *`my-cluster`* con *.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

b. Scegli Aggiorna policy.

AWS CLI

Per creare un ruolo di AWS FargatePod esecuzione con AWS CLI

1. Copia e incolla il contenuto seguente in un file denominato *pod-execution-role-trust-policy*.json. Sostituisci il *codice regionale us-iso-east* con Regione AWS quello in cui si trova il tuo cluster. * Sostituisci *111122223333* con il tuo ID account e *my-cluster* con il nome del cluster. Se vuoi usare lo stesso ruolo per tutti i cluster dell'account, sostituisci *my-cluster* con *.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Condition": {
         "ArnLike": {
            "aws:SourceArn": "arn:aws:eks:region-
code:111122223333:fargateprofile/my-cluster/*"
         }
      },
      "Principal": {
        "Service": "eks-fargate-pods.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Crea un ruolo IAM di esecuzione del Pod.

```
aws iam create-role \
  --role-name AmazonEKSFargatePodExecutionRole \
  --assume-role-policy-document file://"pod-execution-role-trust-policy.json"
```

3. Allegare la policy IAM gestita da Amazon EKS richiesta al ruolo.

```
aws iam attach-role-policy \
  --policy-arn arn:aws:iam::aws:policy/AmazonEKSFargatePodExecutionRolePolicy \
```

```
    --role-name AmazonEKSFargatePodExecutionRole
```

Ruolo IAM di Amazon EKS Connector

Puoi connettere Kubernetes i cluster per visualizzarli nel tuo AWS Management Console. Per collegarti a un cluster Kubernetes, crea un ruolo IAM.

Verificare la presenza di un ruolo del connettore EKS esistente

Per controllare se l'account dispone già di un ruolo di Amazon EKS Connector, utilizzare la procedura indicata di seguito.

Come verificare la presenza di **`AmazonEKSConnectorAgentRole`** nella console IAM

1. Aprire la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Cerca l'elenco dei ruoli per `AmazonEKSConnectorAgentRole`. Se un ruolo che include `AmazonEKSConnectorAgentRole` non esiste, consulta Creazione del ruolo agente di Amazon EKS Connector per crearlo. Se un ruolo che include `AmazonEKSConnectorAgentRole` esiste, seleziona il ruolo per visualizzare le policy allegate.
4. Selezionare Autorizzazioni.
5. Assicurati che la policy gestita da ConnectorAgentPolicy AmazonEks sia associata al ruolo. Se la policy è collegata, il ruolo del connettore Amazon EKS è configurato correttamente.
6. Scegli Trust relationships (Relazioni di attendibilità), quindi scegli Edit trust policy (Modifica policy di attendibilità).
7. Verifica che la relazione di trust includa la policy seguente. Se la relazione di attendibilità corrisponde alla policy seguente, scegli Cancel (Annulla). Se la relazione di attendibilità non corrisponde, copia la policy nella finestra Modifica policy di attendibilità e scegli Aggiorna policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com"
```

```
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Creazione del ruolo agente di Amazon EKS Connector

Puoi usare AWS Management Console o AWS CloudFormation per creare il ruolo di agente del connettore.

AWS CLI

1. Creare un file denominato `eks-connector-agent-trust-policy.json` contenente il seguente JSON da utilizzare per il ruolo IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ssm.amazonaws.com"
                ]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Creare un file denominato `eks-connector-agent-policy.json` contenente il seguente JSON da utilizzare per il ruolo IAM.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "SsmControlChannel",
            "Effect": "Allow",
```

```
            "Action": [
                "ssmmessages:CreateControlChannel"
            ],
            "Resource": "arn:aws:eks:*:*:cluster/*"
        },
        {
            "Sid": "ssmDataplaneOperations",
            "Effect": "Allow",
            "Action": [
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenDataChannel",
                "ssmmessages:OpenControlChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

3. Crea il ruolo agente di Amazon EKS Connector utilizzando la policy di attendibilità e la policy creata negli elementi precedenti dell'elenco.

```
aws iam create-role \
     --role-name AmazonEKSConnectorAgentRole \
     --assume-role-policy-document file://eks-connector-agent-trust-policy.json
```

4. Allegare la policy al ruolo agente di Amazon EKS Connector.

```
aws iam put-role-policy \
     --role-name AmazonEKSConnectorAgentRole \
     --policy-name AmazonEKSConnectorAgentPolicy \
     --policy-document file://eks-connector-agent-policy.json
```

AWS CloudFormation

Per creare il tuo ruolo di agente del connettore Amazon EKS con AWS CloudFormation.

1. Salva il seguente AWS CloudFormation modello in un file di testo sul tuo sistema locale.

> **Note**
>
> Questo modello crea anche il ruolo collegato al servizio che altrimenti verrebbe stato creato al momento della chiamata API `registerCluster`. Per informazioni dettagliate, vedi Utilizzo di ruoli per connettere un cluster Kubernetes ad Amazon EKS.

```
---
AWSTemplateFormatVersion: '2010-09-09'
Description: 'Provisions necessary resources needed to register clusters in EKS'
Parameters: {}
Resources:
  EKSConnectorSLR:
    Type: AWS::IAM::ServiceLinkedRole
    Properties:
      AWSServiceName: eks-connector.amazonaws.com

  EKSConnectorAgentRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: Allow
            Action: [ 'sts:AssumeRole' ]
            Principal:
              Service: 'ssm.amazonaws.com'

  EKSConnectorAgentPolicy:
    Type: AWS::IAM::Policy
    Properties:
      PolicyName: EKSConnectorAgentPolicy
      Roles:
        - {Ref: 'EKSConnectorAgentRole'}
      PolicyDocument:
        Version: '2012-10-17'
        Statement:
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateControlChannel' ]
            Resource:
            - Fn::Sub: 'arn:${AWS::Partition}:eks:*:*:cluster/*'
```

```
          - Effect: 'Allow'
            Action: [ 'ssmmessages:CreateDataChannel',
 'ssmmessages:OpenDataChannel', 'ssmmessages:OpenControlChannel' ]
            Resource: "*"
Outputs:
  EKSConnectorAgentRoleArn:
    Description: The agent role that EKS connector uses to communicate with
 Servizi AWS.
    Value: !GetAtt EKSConnectorAgentRole.Arn
```

2. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.
3. Scegliere Crea pila (con nuove risorse o risorse esistenti).
4. In Specify template (Specifica modello), selezionare Upload a template file (Carica un file di modello) e Choose file (Scegli file).
5. Scegliere il file creato in precedenza, quindi selezionare Next (Successivo).
6. Per Stack name (Nome pila), immettere un nome per il ruolo, ad esempio `eksConnectorAgentRole`, quindi scegliere Next (Successivo).
7. Nella pagina Configure stack options (Configura opzioni pila), scegliere Next (Successivo).
8. Nella pagina Revisione, esaminare le informazioni, accettare che la pila può creare risorse IAM, quindi scegliere Crea pila.

AWS politiche gestite per Amazon Elastic Kubernetes Service

Una politica AWS gestita è una politica autonoma creata e amministrata da AWS. AWS le politiche gestite sono progettate per fornire autorizzazioni per molti casi d'uso comuni, in modo da poter iniziare ad assegnare autorizzazioni a utenti, gruppi e ruoli.

Tieni presente che le policy AWS gestite potrebbero non concedere le autorizzazioni con il privilegio minimo per i tuoi casi d'uso specifici, poiché sono disponibili per tutti i clienti. AWS Consigliamo pertanto di ridurre ulteriormente le autorizzazioni definendo policy gestite dal cliente specifiche per i tuoi casi d'uso.

Non è possibile modificare le autorizzazioni definite nelle politiche gestite. AWS Se AWS aggiorna le autorizzazioni definite in una politica AWS gestita, l'aggiornamento ha effetto su tutte le identità principali (utenti, gruppi e ruoli) a cui è associata la politica. AWS è più probabile che aggiorni una

policy AWS gestita quando ne Servizio AWS viene lanciata una nuova o quando diventano disponibili nuove operazioni API per i servizi esistenti.

Per ulteriori informazioni, consultare Policy gestite da AWSnella Guida per l'utente di IAM.

AWS politica gestita: Amazoneks_CNI_Policy

É possibile allegare la `AmazonEKS_CNI_Policy` alle entità IAM. Prima di creare un gruppo di nodi Amazon EC2, questa policy deve essere allegata al ruolo IAM del nodo o a un ruolo IAM che viene utilizzato in modo specifico dal Amazon VPC CNI plugin for Kubernetes. In questo modo può eseguire operazioni per conto dell'utente. Si consiglia di allegare le policy a un ruolo utilizzato solo dal plugin. Per ulteriori informazioni, consulta Utilizzo del componente aggiuntivo Amazon VPC CNI plugin for Kubernetes di Amazon EKS e Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le attività seguenti:

- **`ec2:*NetworkInterface`**e **`ec2:*PrivateIpAddresses`** — Consente al plug-in Amazon VPC CNI di eseguire azioni come il provisioning di interfacce di rete elastiche e indirizzi IP per fornire reti Pods per le applicazioni eseguite in Amazon EKS.
- **`ec2`**azioni di lettura: consente al plug-in Amazon VPC CNI di eseguire azioni come descrivere istanze e sottoreti per visualizzare la quantità di indirizzi IP gratuiti nelle sottoreti Amazon VPC. Il VPC CNI può utilizzare gli indirizzi IP gratuiti in ogni sottorete per scegliere le sottoreti con gli indirizzi IP più liberi da utilizzare durante la creazione di un'interfaccia di rete elastica.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta Amazoneks_CNI_Policy nella Managed Policy Reference Guide. AWS

AWS politica gestita: AmazonEks ClusterPolicy

È possibile allegare `AmazonEKSClusterPolicy` alle entità IAM. Prima di creare un cluster, è necessario disporre di un ruolo IAM del cluster con questa policy allegata. Kubernetesi cluster gestiti da Amazon EKS effettuano chiamate verso altri AWS servizi per tuo conto. Ciò serve a gestire le risorse utilizzate con il servizio.

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le attività seguenti:

- **`autoscaling`**: leggi e aggiorna la configurazione di un gruppo Auto Scaling. Queste autorizzazioni non vengono utilizzate da Amazon EKS, ma rimangono nella policy per la compatibilità con le versioni precedenti.
- **`ec2`**: lavora con volumi e risorse di rete associati ai nodi Amazon EC2. Ciò è necessario affinché il piano di controllo (control-plane) Kubernetes possa unire le istanze a un cluster ed eseguire dinamicamente il provisioning e la gestione dei volumi Amazon EBS richiesti dai volumi persistenti di Kubernetes.
- **`elasticloadbalancing`**: lavora con Elastic Load Balancers e aggiungi nodi come destinazioni. Ciò è necessario affinché il piano di controllo Kubernetespossa eseguire dinamicamente il provisioning degli Elastic Load Balancer richiesti dai servizi Kubernetes.
- **`iam`**: creazione di un ruolo collegato ai servizi. Ciò è necessario affinché il piano di controllo Kubernetespossa eseguire dinamicamente il provisioning degli Elastic Load Balancer richiesti dai servizi Kubernetes.
- **`kms`**:leggi una chiave da AWS KMS. Questa attività è necessaria affinché il piano di controllo (control-plane) Kubernetes supporti la crittografia dei segreti di Kubernetes archiviati in `etcd`.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta ClusterPolicyAmazonEks nella AWS Managed Policy Reference Guide.

AWS politica gestita: AmazonEks FargatePodExecutionRolePolicy

È possibile allegare `AmazonEKSFargatePodExecutionRolePolicy` alle entità IAM. Prima di poter creare un profilo di Fargate, è necessario creare un ruolo di esecuzione del Pod Fargate e allegare questa policy ad esso. Per ulteriori informazioni, consulta Creazione di un ruolo di esecuzione del Pod Fargate e AWS Fargate profilo.

Questa politica concede al ruolo le autorizzazioni che forniscono l'accesso ad altre risorse di AWS servizio necessarie per eseguire Amazon EKS su Pods Fargate.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le attività seguenti:

- **`ecr`**: consente ai pod in esecuzione su Fargate di estrarre le immagini del container memorizzate in Amazon ECR.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta FargatePodExecutionRolePolicyAmazonEks nella AWS Managed Policy Reference Guide.

AWS politica gestita: AmazonEks ForFargateServiceRolePolicy

Non è possibile allegare `AmazonEKSForFargateServiceRolePolicy` alle entità IAM. Questa policy è allegata a un ruolo collegato ai servizi che consente ad Amazon EKS di eseguire operazioni per conto dell'utente. Per ulteriori informazioni, consulta. AWSServiceRoleforAmazonEKSForFargate

Questa policy concede ad Amazon EKS le autorizzazioni necessarie per eseguire le attività di Fargate. La policy viene utilizzato solo se si dispone di nodi Fargate.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le seguenti attività.

- **`ec2`**: crea ed elimina interfacce di rete elastiche e descrive le interfacce di rete elastiche e le risorse. Questa operazione è necessaria affinché il servizio Amazon EKS Fargate possa configurare la rete VPC necessaria per i pod Fargate.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta ForFargateServiceRolePolicyAmazonEks nella AWS Managed Policy Reference Guide.

AWS politica gestita: AmazonEks ServicePolicy

È possibile allegare `AmazonEKSServicePolicy` alle entità IAM. I cluster creati prima del 16 aprile 2020 richiedono all'utente di creare un ruolo IAM e allegarvi questa policy. I cluster creati a partire dal 16 aprile 2020 non richiedono la creazione di un ruolo e non richiedono l'assegnazione di questa policy. Quando crei un cluster utilizzando un principale IAM che dispone dell'`iam:CreateServiceLinkedRole`autorizzazione, il ruolo AWS ServiceRoleforAmazonEKS collegato al servizio viene creato automaticamente per te. Il ruolo collegato al servizio ha la proprietà AWS politica gestita: AmazonEks ServiceRolePolicy ad esso allegata.

Questa policy consente ad Amazon EKS di creare e gestire le risorse necessarie per gestire i cluster Amazon EKS.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le seguenti attività.

- **`eks`**: aggiorna la versione Kubernetes del cluster dopo aver avviato un aggiornamento. Questa autorizzazione non viene utilizzata da Amazon EKS, ma rimane nella policy per la compatibilità con le versioni precedenti.
- **`ec2`**: lavora con le interfacce di rete elastiche e altre risorse e tag di rete. Ciò è richiesto da Amazon EKS per configurare la rete che facilita la comunicazione tra i nodi e il piano di controllo Kubernetes.
- **`route53`**: associa un VPC a una zona ospitata. Ciò è richiesto da Amazon EKS per abilitare la rete di endpoint privati per il server API del cluster Kubernetes.
- **`logs`**: log eventi. Ciò è necessario per consentire ad Amazon EKS di spedire i log Kubernetes del piano di controllo a CloudWatch.
- **`iam`**: creazione di un ruolo collegato ai servizi. Questa operazione è necessaria affinché Amazon EKS possa creare il ruolo collegato al servizio `AWSServiceRoleForAmazonEKS` per conto dell'utente.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta ServicePolicyAmazonEks nella AWS Managed Policy Reference Guide.

AWS politica gestita: AmazonEks ServiceRolePolicy

Non è possibile allegare `AmazonEKSServiceRolePolicy` alle entità IAM. Questa policy è allegata a un ruolo collegato ai servizi che consente ad Amazon EKS di eseguire operazioni per conto dell'utente. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi per Amazon EKS. Quando crei un cluster utilizzando un principale IAM che dispone dell'`iam:CreateServiceLinkedRole`autorizzazione, il ruolo AWS ServiceRoleforAmazonEKS collegato al servizio viene creato automaticamente per te e questa policy gli viene allegata.

Questa policy consente al ruolo collegato al servizio di chiamare i AWS servizi per tuo conto.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le seguenti attività.

- **`ec2`**: crea e descrive le interfacce di rete elastiche e le istanze di Amazon EC2, il gruppo di sicurezza del cluster e i VPC necessari per la creazione del cluster.
- **`iam`**: elenca tutti i criteri gestiti allegati a un ruolo IAM. Ciò è necessario affinché Amazon EKS possa pubblicare e convalidare tutte le politiche gestite e le autorizzazioni necessarie per la creazione di un cluster.
- Associazione di un VPC a una zona ospitata: richiesto da Amazon EKS per abilitare la rete di endpoint privati per il server API del cluster Kubernetes.
- Evento di registro: è necessario per consentire ad Amazon EKS di inviare i log Kubernetes del piano di controllo a CloudWatch.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta ServiceRolePolicyAmazonEks nella AWS Managed Policy Reference Guide.

AWS politica gestita: AmazonekSVPC ResourceController

È possibile allegare la policy `AmazonEKSVPCResourceController`alle identità IAM. Se utilizzi gruppi di sicurezza per Pods, dovrai allegare questa policy al Ruolo IAM del cluster Amazon EKS perché possa eseguire operazioni per tuo conto.

Questa policy concede le autorizzazioni del ruolo cluster per gestire le interfacce di rete elastiche e gli indirizzi IP per i nodi.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le attività seguenti:

- **`ec2`**: gestisci le interfacce di rete elastiche e gli indirizzi IP per supportare i gruppi di sicurezza dei Pod e i nodi Windows.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta ResourceControllerAmazonekSVPC nella Managed Policy Reference Guide. AWS

AWS politica gestita: AmazonEks WorkerNodePolicy

É possibile allegare la `AmazonEKSWorkerNodePolicy` alle entità IAM. È necessario allegare questa policy a un Ruolo IAM del nodo che va specificato quando si creano nodi Amazon EC2 che consentono ad Amazon EKS di eseguire operazioni per conto dell'utente. Se si crea un gruppo di

nodi utilizzando `eksctl`, si crea il ruolo IAM del nodo e si allega automaticamente questa policy al ruolo.

Questa policy concede ai nodi Amazon EKS Amazon EC2 le autorizzazioni per connettersi ai cluster Amazon EKS.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le attività seguenti:

- **`ec2`**: leggere le informazioni sul volume dell'istanza e sulla rete. Ciò è necessario affinché i nodi Kubernetes possano descrivere le informazioni sulle risorse Amazon EC2 necessarie per l'unione del nodo al cluster Amazon EKS.
- **`eks`**: descrivere in modo facoltativo il cluster come parte del bootstrap del nodo.
- **`eks-auth:AssumeRoleForPodIdentity`**: consentire il recupero delle credenziali per i carichi di lavoro EKS sul nodo. Ciò è necessario per il funzionamento di EKS Pod Identity.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta WorkerNodePolicyAmazonEks nella AWS Managed Policy Reference Guide.

AWS politica gestita: AWSServiceRoleForAmazonEKSNodegroup

Non è possibile allegare `AWS ServiceRoleForAmazonEKSNodegroup` alle entità IAM. Questa policy è allegata a un ruolo collegato ai servizi che consente ad Amazon EKS di eseguire operazioni per conto dell'utente. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi per Amazon EKS.

Questa policy concede l'autorizzazione ai ruoli `AWS ServiceRoleForAmazonEKSNodegroup` che consentono di creare e gestire gruppi di nodi Amazon EC2 nell'account.

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni che consentono ad Amazon EKS di completare le attività seguenti:

- **`ec2`**: lavora con gruppi di sicurezza, tag e modelli di avvio. Ciò è necessario per i gruppi di nodi gestiti da Amazon EKS per abilitare la configurazione dell'accesso remoto. Inoltre, i gruppi di nodi gestiti da Amazon EKS creano un modello di avvio per conto dell'utente. Questo consente di configurare il gruppo Amazon EC2 Auto Scaling che supporta ogni gruppo di nodi gestito.

- **`iam`**: creazione di un ruolo collegato ai servizi e passa un ruolo. Ciò è richiesto dai gruppi di nodi gestiti da Amazon EKS per gestire i profili di istanza per il ruolo passato durante la creazione di un gruppo di nodi gestito. Questo profilo dell'istanza viene utilizzato dalle istanze Amazon EC2 avviate come parte di un gruppo di nodi gestito. Amazon EKS deve creare ruoli collegati al servizio per altri servizi come, ad esempio i gruppi di Amazon EC2 Auto Scaling. Queste autorizzazioni vengono utilizzate nella creazione di un gruppo di nodi gestito.
- **`autoscaling`**: lavora con gruppi Auto Scaling di sicurezza. Questo è richiesto dai gruppi di nodi gestiti da Amazon EKS per gestire il gruppo di Amazon EC2 Auto Scaling che supporta ciascun gruppo di nodi gestito. Viene inoltre utilizzato per supportare funzionalità quali la rimozione di Pods quando i nodi vengono terminati o riavviati durante gli aggiornamenti dei gruppi di nodi.

Per visualizzare l'ultima versione del documento sulla policy JSON, consulta la AWS Managed Policy AWSServiceRoleForAmazonEKSNodegroupReference Guide.

AWS politica gestita: AmazoneBSCSI DriverPolicy

La policy `AmazonEBSCSIDriverPolicy` consente al driver Container Storage Interface (CSI) di Amazon EBS di creare, modificare, collegare, scollegare ed eliminare volumi per conto dell'utente. Concede inoltre al driver CSI di EBS le autorizzazioni necessarie per creare snapshot, eliminarli e per elencare istanze, volumi e snapshot.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta DriverServiceRolePolicyAmazoneBSCSI nella Managed Policy Reference Guide. AWS

AWS politica gestita: AmazoneFSCSI DriverPolicy

La`AmazonEFSCSIDriverPolicy`la policy consente all'Amazon EFS Container Storage Interface (CSI) di creare ed eliminare punti di accesso per tuo conto. Concede inoltre al driver CSI di Amazon EFS le autorizzazioni per elencare i punti di accesso, i file system, gli obiettivi di montaggio e le zone di disponibilità di Amazon EC2.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta DriverServiceRolePolicyAmazoneFSCSI nella Managed Policy Reference Guide. AWS

AWS politica gestita: AmazonEks LocalOutpostClusterPolicy

È possibile collegare questa policy alle entità IAM. Prima di creare un cluster locale, devi collegare questa politica al tuo ruolo di cluster. Kubernetesi cluster gestiti da Amazon EKS effettuano chiamate verso altri AWS servizi per tuo conto. Ciò serve a gestire le risorse utilizzate con il servizio.

La `AmazonEKSLocalOutpostClusterPolicy` include le autorizzazioni seguenti:

- **`ec2`**: autorizzazioni necessarie affinché le istanze Amazon EC2 si uniscano correttamente al cluster come istanze del piano di controllo.
- **`ssm`**: consente la connessione di Amazon EC2 Systems Manager all'istanza del piano di controllo, utilizzata da Amazon EKS per comunicare e gestire il cluster locale nel tuo account.
- **`logs`**— Consente alle istanze di inviare i log ad Amazon. CloudWatch
- **`secretsmanager`**— Consente alle istanze di ottenere ed eliminare i dati di bootstrap per le istanze del piano di controllo in modo sicuro. AWS Secrets Manager
- **`ecr`**: consente a Pods e ai container in esecuzione sulle istanze del piano di controllo di estrarre le immagini di container memorizzate in Amazon Elastic Container Registry.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta LocalOutpostClusterPolicyAmazonEks nella AWS Managed Policy Reference Guide.

AWS politica gestita: AmazonEks LocalOutpostServiceRolePolicy

Non è possibile attribuire questa policy alle entità IAM. Quando si crea un cluster tramite un principale IAM che dispone dell'autorizzazione `iam:CreateServiceLinkedRole`, Amazon EKS crea automaticamente il ruolo collegato al servizio `AWSServiceRoleforAmazonEKSLocalOutpost` per conto tuo e questa policy è collegata ad esso. Questa politica consente al ruolo collegato al servizio di chiamare AWS i servizi per tuo conto per i cluster locali.

La `AmazonEKSLocalOutpostServiceRolePolicy` include le autorizzazioni seguenti:

- **`ec2`**: consente ad Amazon EKS di utilizzare le risorse di sicurezza, rete e altro tipo per avviare e gestire correttamente le istanze del piano di controllo nel tuo account.
- **`ssm`**: consente la connessione di Amazon EC2 Systems Manager alle istanze del piano di controllo, utilizzata da Amazon EKS per comunicare e gestire il cluster locale nel tuo account.
- **`iam`**: consente ad Amazon EKS di gestire il profilo dell'istanza associato alle istanze del piano di controllo.
- **`secretsmanager`**— Consente ad Amazon EKS di inserire i dati di bootstrap per le istanze del piano di controllo in AWS Secrets Manager modo che possano essere referenziati in modo sicuro durante il bootstrap delle istanze.
- **`outposts`**: consente ad Amazon EKS di ottenere le informazioni sull'Outpost dal tuo account per avviare correttamente un cluster locale in un Outpost.

Per visualizzare la versione più recente del documento sulla policy JSON, consulta LocalOutpostServiceRolePolicyAmazonEks nella AWS Managed Policy Reference Guide.

Amazon EKS si aggiorna alle politiche AWS gestite

Visualizza i dettagli sugli aggiornamenti delle politiche AWS gestite per Amazon EKS da quando questo servizio ha iniziato a tracciare queste modifiche. Per gli avvisi automatici sulle modifiche apportate a questa pagina, effettua l'abbonamento al feed RSS nella pagina della cronologia dei documenti di Amazon EKS.

Modifica	Descrizione	Data
Amazoneks_CNI_Policy — Aggiornamento a una politica esistente	Amazon EKS ha aggiunto nuove `ec2:DescribeSubnets` autorizza zioni per consentire loro di Amazon VPC CNI plugin for Kubernetes visualizzare la quantità di indirizzi IP gratuiti nelle sottoreti Amazon VPC. Il VPC CNI può utilizzare gli indirizzi IP gratuiti in ogni sottorete per scegliere le sottoreti con gli indirizzi IP più liberi da utilizzare durante la creazione di un'interfaccia di rete elastica.	4 marzo 2024
WorkerNodePolicyAmazonEks — Aggiornamento a una politica esistente	Amazon EKS ha aggiunto nuove autorizzazioni per consentire le associazioni EKS Pod Identity. Amazon EKS Pod Identity Agent utilizza il ruolo del nodo.	26 novembre 2023
Introduzione di Amazon FSCSI. DriverPolicy	AWS ha introdotto il`AmazonEFS CSIDriverPolicy` .	26 luglio 2023

Modifica	Descrizione	Data
Sono state aggiunte le autorizzazioni ad AmazonEksClusterPolicy.	È stata aggiunta l'autorizzazione `ec2:DescribeAvailabilityZones` per consentire ad Amazon EKS di ottenere i dettagli AZ durante l'individuazione automatica delle sottoreti nella creazione di sistemi di bilanciamento del carico.	7 febbraio 2023
Condizioni politiche aggiornate in AmazoneBSCSI. DriverPolicy	Sono state rimosse condizioni della policy non valide con caratteri jolly nel campo chiave `StringLike` . È stata aggiunta, inoltre, una nuova condizione `ec2:ResourceTag/kubernetes.io/created-for/pvc/name: "*"` a `ec2:DeleteVolume` che consente al driver CSI EBS di eliminare volumi creati dal plugin nella struttura.	17 novembre 2022
Sono state aggiunte le autorizzazioni ad AmazonEksLocalOutpostServiceRolePolicy.	Aggiunti `ec2:DescribeVPCAttribute` , `ec2:GetConsoleOutput` e `ec2:DescribeSecret` per consentire una migliore convalida dei prerequisiti e il controllo gestito del ciclo di vita. Sono stati inoltre aggiunti `ec2:DescribePlacementGroups` , `"arn:aws:ec2:*:*:placement-group/*"` e `ec2:RunInstances` per supportare il controllo del posizionamento delle istanze Amazon EC2 del piano di controllo su Outposts.	24 ottobre 2022

Modifica	Descrizione	Data
Aggiorna le autorizzazioni di Amazon Elastic Container Registry in AmazoneKS LocalOutpostClusterPolicy.	L'azione `ecr:GetDownloadUrl ForLayer` è stata spostata da tutte le sezioni delle risorse a una sezione con ambito. Aggiunta la risorsa `arn:aws:ecr:*:*:repository/eks/*` . Risorsa `arn:aws:ecr:*:*:repository/eks/eks-certificates-controller-public` rimossa. Questa risorsa è coperta dalla risorsa `arn:aws:ecr:*:*:repository/eks/*` aggiunta.	20 ottobre 2022
Sono state aggiunte le autorizzazioni ad AmazonEks LocalOutpostClusterPolicy.	È stato aggiunto il repository `arn:aws:ecr:*:*:repository/kubelet-config-updater` Amazon Elastic Container Registry in modo che le istanze del piano di controllo del cluster possano aggiornare alcuni argomenti `kubelet`.	31 agosto 2022
Presentazione di AmazonEks LocalOutpostClusterPolicy.	AWS ha introdotto il. `AmazonEKS LocalOutpostClusterPolicy`	24 agosto 2022
Presentazione di AmazonEks LocalOutpostServiceRolePolicy.	AWS ha introdotto il. `AmazonEKS LocalOutpostServiceRolePolicy`	23 agosto 2022
Ha introdotto Amazon DriverPolicy BSCSI.	AWS ha introdotto il`AmazonEBS CSIDriverPolicy` .	4 aprile 2022
Sono state aggiunte le autorizzazioni ad AmazonEks WorkerNodePolicy.	È stato aggiunto `ec2:DescribeInstanceTypes` per consentire alle AMI ottimizzate per Amazon EKS di rilevare automaticamente le proprietà a livello di istanza.	21 marzo 2022

Modifica	Descrizione	Data
Autorizzazioni aggiunte a. AWSServiceRoleForAmazonEKSNodegroup	Aggiunta l'autorizzazione `autoscaling:EnableMetricsCollection` per consentire ad Amazon EKS l'abilitazione della raccolta di parametri.	13 dicembre 2021
Sono state aggiunte le autorizzazioni ad AmazonEksClusterPolicy.	Aggiunta delle autorizzazioni `ec2:DescribeAccountAttributes` , `ec2:DescribeAddresses` , e `ec2:DescribeInternetGateways` per consentire ad Amazon EKS di creare un ruolo collegato al servizio per un Network Load Balancer (load balancer di Rete).	17 giugno 2021
Amazon EKS ha iniziato a monitorare le modifiche	Amazon EKS ha iniziato a tracciare le modifiche per le sue politiche AWS gestite.	17 giugno 2021

Risoluzione dei problemi di IAM

Questo argomento illustra alcuni errori comuni che si potrebbero verificare durante l'utilizzo di Amazon EKS con IAM, e il modo in cui gestirli.

AccessDeniedException

Se ricevi un messaggio `AccessDeniedException` quando chiami un'operazione AWS API, le credenziali principali IAM che stai utilizzando non dispongono delle autorizzazioni necessarie per effettuare quella chiamata.

```
An error occurred (AccessDeniedException) when calling the DescribeCluster operation:
User: arn:aws:iam::111122223333:user/user_name is not authorized to perform:
eks:DescribeCluster on resource: arn:aws:eks:region:111122223333:cluster/my-cluster
```

Nel messaggio di esempio precedente, l'utente non dispone delle autorizzazioni per chiamare l'operazione API `DescribeCluster` di Amazon EKS. Per fornire le autorizzazioni da amministratore Amazon EKS a un principale IAM, consultare Esempi di policy basate su identità Amazon EKS.

Per informazioni generali su IAM, consultare Controllo dell'accesso tramite policy nella Guida per l'utente di IMA.

Impossibile visualizzare i nodi nella scheda Calcolo (o qualsiasi altro elemento nella scheda Risorse) e si riceve un errore nella AWS Management Console

È possibile che venga visualizzato un messaggio di errore della console che recita `Your current user or role does not have access to Kubernetes objects on this EKS cluster`. Assicurati che l'utente principale IAM AWS Management Console con cui stai utilizzando disponga delle autorizzazioni necessarie. Per ulteriori informazioni, consulta Autorizzazioni richieste.

La **`ConfigMap`** per aws-auth non concede l'accesso al cluster

L'Autenticatore AWS IAM non consente un percorso nell'ARN del ruolo utilizzato nella `ConfigMap`. Pertanto, rimuovi il percorso prima di specificare `rolearn`. Ad esempio, modifica `arn:aws:iam::111122223333:role/team/developers/eks-admin` in `arn:aws:iam::111122223333:role/eks-admin`.

Non sono autorizzato a eseguire iam: PassRole

Se si riceve un errore che indica che non si è autorizzati a eseguire l'operazione `iam:PassRole`, è necessario aggiornare le policy per poter passare un ruolo ad Amazon EKS.

Alcuni Servizi AWS consentono di passare un ruolo esistente a quel servizio invece di creare un nuovo ruolo di servizio o un ruolo collegato al servizio. Per eseguire questa operazione, è necessario disporre delle autorizzazioni per trasmettere il ruolo al servizio.

L'errore di esempio seguente si verifica quando un utente IAM denominato `marymajor` cerca di utilizzare la console per eseguire un'operazione in Amazon EKS. Tuttavia, l'azione richiede che il servizio disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone delle autorizzazioni per passare il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In questo caso, le policy di Mary devono essere aggiornate per poter eseguire l'operazione `iam:PassRole`.

Se hai bisogno di aiuto, contatta il tuo AWS amministratore. L'amministratore è la persona che ti ha fornito le credenziali di accesso.

Voglio consentire a persone esterne al mio AWS account di accedere alle mie risorse Amazon EKS

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per servizi che supportano policy basate su risorse o liste di controllo accessi (ACL), utilizza tali policy per concedere alle persone l'accesso alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se Amazon EKS supporta queste caratteristiche, consultare Funzionamento di Amazon EKS con IAM.
- Per scoprire come fornire l'accesso alle tue risorse attraverso Account AWS le risorse di tua proprietà, consulta Fornire l'accesso a un utente IAM in un altro Account AWS di tua proprietà nella IAM User Guide.
- Per scoprire come fornire l'accesso alle tue risorse a terze parti Account AWS, consulta Fornire l'accesso a soggetti Account AWS di proprietà di terze parti nella Guida per l'utente IAM.
- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta Fornire l'accesso a utenti autenticati esternamente (Federazione delle identità) nella Guida per l'utente di IAM.
- Per informazioni sulle differenze tra l'utilizzo di ruoli e policy basate su risorse per l'accesso multi-account, consulta Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente IAM.

I container dei pod riceveranno il seguente errore: **`An error occurred (SignatureDoesNotMatch) when calling the GetCallerIdentity operation: Credential should be scoped to a valid region`**

I tuoi contenitori ricevono questo errore se l'applicazione effettua esplicitamente richieste all'endpoint AWS STS globale (`https://sts.amazonaws`) e il tuo account di Kubernetes servizio è configurato per utilizzare un endpoint regionale. Puoi risolvere il problema con una delle opzioni seguenti:

- Aggiorna il codice dell'applicazione per rimuovere le chiamate esplicite all'endpoint globale. AWS STS
- Aggiorna il codice dell'applicazione per effettuare chiamate esplicite agli endpoint regionali, ad esempio `https://sts.us-west-2.amazonaws.com`. L'applicazione deve avere una

ridondanza integrata per scegliere una Regione AWS diversa in caso di fallimento del servizio nella Regione AWS. Per ulteriori informazioni, consulta la sezione Gestione di AWS STS in una Regione AWS nella Guida per l'utente di IAM.

- Configura gli account del servizio per l'utilizzo dell'endpoint globale. Per impostazione predefinita, tutti i cluster con versioni precedenti alla `1.22` utilizzavano l'endpoint globale, mentre quelli successivi alla versione `1.22` utilizzano l'endpoint regionale. Per ulteriori informazioni, consulta Configurare l' AWS Security Token Service endpoint per un account di servizio.

Ruoli e utenti predefiniti Kubernetes creati da Amazon EKS

Quando crei un cluster Kubernetes, su tale cluster vengono create diverse identità Kubernetes predefinite per il corretto funzionamento di Kubernetes. Amazon EKS crea identità Kubernetes per ciascuno dei suoi componenti predefiniti. Le identità forniscono un controllo delle autorizzazioni basato sui ruoli (RBAC) Kubernetes per i componenti del cluster. Per ulteriori informazioni, consulta Utilizzo dell'autorizzazione RBAC nella documentazione di Kubernetes.

Quando installi componenti aggiuntivi opzionali nel tuo cluster, è possibile che vengano aggiunte identità supplementari Kubernetes al cluster. Per ulteriori informazioni sulle identità non trattate in questo argomento, consulta la documentazione del componente aggiuntivo.

Puoi visualizzare l'elenco delle identità Kubernetes create da Amazon EKS sul tuo cluster utilizzando lo strumento a riga di comando della AWS Management Console o `kubectl`. Tutte le identità degli utenti vengono visualizzate nei registri di `kube` controllo disponibili tramite Amazon. CloudWatch

AWS Management Console

Prerequisito

Il principale IAM che devi utilizzare deve disporre delle autorizzazioni descritte in Autorizzazioni richieste.

Per visualizzare le identità create da Amazon EKS utilizzando AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nell'elenco Cluster, seleziona il cluster contenente le identità da visualizzare.
3. Scegliere la scheda Resources (Risorse).
4. In Resource types (Tipi di risorse), scegli Authorization (Autorizzazione).

5. Scegli, ClusterRoles, ClusterRoleBindings, Ruoli o RoleBindings. Tutte le risorse con il prefisso eks sono create da Amazon EKS. Le risorse di identità supplementari create da Amazon EKS sono:

   - La ClusterRolee ClusterRoleBindingdenominata aws-node. Le risorse aws-node supportano il Amazon VPC CNI plugin for Kubernetes, che Amazon EKS installa su tutti i cluster.
   - Un ClusterRolenome vpc-resource-controller-rolee un ClusterRoleBindingnome. vpc-resource-controller-rolebinding Queste risorse supportano il Amazon VPC resource controller (Controller risorse Amazon VPC), che Amazon EKS installa su tutti i cluster.

   Oltre alle risorse visualizzate nella console, nel cluster esistono le seguenti identità utente speciali, anche se non sono visibili nella configurazione del cluster:

   - **`eks:cluster-bootstrap`**: utilizzato per operazioni `kubectl` durante il bootstrap del cluster.
   - **`eks:support-engineer`**: utilizzato per le operazioni di gestione del cluster.
6. Scegli una risorsa specifica per visualizzarne i dettagli. Per impostazione predefinita, le informazioni sono mostrate in Structured view (Visualizzazione strutturata). Nell'angolo superiore destro della pagina dei dettagli puoi scegliere la Raw view (Visualizzazione non elaborata) per vedere tutte le informazioni per la risorsa.

Kubectl

Prerequisito

L'entità che utilizzi (AWS Identity and Access Management (IAM) o OpenID Connect (OIDC)) per elencare le Kubernetes risorse sul cluster deve essere autenticata da IAM o dal tuo provider di OIDC identità. All'entità devono essere concesse le autorizzazioni per utilizzare i verbi `get` e `list` di Kubernetes per le risorse `Role`, `ClusterRole`, `RoleBinding` e `ClusterRoleBinding` nel cluster che devono essere utilizzate dall'entità. Per ulteriori informazioni sulla concessione alle entità IAM dell'accesso al tuo cluster, consulta the section called “Concedi l'accesso alle API Kubernetes”. Per ulteriori informazioni sulla concessione alle entità autenticate dal tuo provider OIDC dell'accesso al tuo cluster, consulta Autentica gli utenti del tuo cluster da un provider di OpenID Connect identità.

Come visualizzare le identità create da Amazon EKS tramite **`kubectl`**

Esegui il comando per il tipo di risorsa da visualizzare. Tutte le risorse restituite con il prefisso eks sono create da Amazon EKS. Oltre alle risorse restituite nell'output dai comandi, nel cluster esistono le seguenti identità utente speciali, anche se non sono visibili nella configurazione del cluster:

- **`eks:cluster-bootstrap`** – Utilizzato per operazioni `kubectl` durante il bootstrap del cluster.
- **`eks:support-engineer`**: utilizzato per le operazioni di gestione del cluster.

ClusterRoles— `ClusterRoles` sono limitati al cluster, quindi qualsiasi autorizzazione concessa a un ruolo si applica alle risorse in qualsiasi spazio dei Kubernetes nomi del cluster.

Il comando seguente restituisce tutti i `ClusterRoles` di Kubernetes creati da Amazon EKS nel tuo cluster.

```
kubectl get clusterroles | grep eks
```

Oltre ai `ClusterRoles` restituiti nell'output con prefisso, esistono i seguenti `ClusterRoles`.

- **`aws-node`**: questo `ClusterRole` supporta il Amazon VPC CNI plugin for Kubernetes, che Amazon EKS installa su tutti i cluster.
- **`vpc-resource-controller-role`**: questo `ClusterRole` supporta il Amazon VPC resource controller (Controller risorse Amazon VPC), che Amazon EKS installa su tutti i cluster.

Per visualizzare le specifiche di un `ClusterRole`, sostituisci *`eks:k8s-metrics`* nel comando seguente con un `ClusterRole` restituito nell'output del comando precedente. L'esempio seguente restituisce la specifica per `ClusterRole` *`eks:k8s-metrics`*.

```
kubectl describe clusterrole eks:k8s-metrics
```

Di seguito viene riportato un output di esempio:

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names  Verbs
```

```
---------          -----------------  --------------  -----
                   [/metrics]         []              [get]
endpoints          []                 []              [list]
nodes              []                 []              [list]
pods               []                 []              [list]
deployments.apps   []                 []              [list]
```

ClusterRoleBindings— `ClusterRoleBindings` sono limitati al cluster.

Il comando seguente restituisce tutti i `ClusterRoleBindings` Kubernetes creati da Amazon EKS nel tuo cluster.

```
kubectl get clusterrolebindings | grep eks
```

Oltre ai `ClusterRoleBindings` restituiti nell'output, esistono i seguenti `ClusterRoleBindings`.

- **`aws-node`** – `ClusterRoleBinding` Supporta il Amazon VPC CNI plugin for Kubernetes, che Amazon EKS installa su tutti i cluster.
- **`vpc-resource-controller-rolebinding`**: questo `ClusterRoleBinding` supporta il Amazon VPC resource controller (Controller risorse Amazon VPC), che Amazon EKS installa su tutti i cluster.

Per visualizzare le specifiche di un `ClusterRoleBinding`, sostituisci *`eks:k8s-metrics`* nel comando seguente con un `ClusterRoleBinding` restituito nell'output del comando precedente. L'esempio seguente restituisce la specifica per `ClusterRoleBinding` *`eks:k8s-metrics`*.

```
kubectl describe clusterrolebinding eks:k8s-metrics
```

Di seguito viene riportato un output di esempio:

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  ClusterRole
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
```

```
  User  eks:k8s-metrics
```

Roles (Ruoli): i `Roles` sono racchiusi in uno spazio dei nomi Kubernetes. Tutti i `Roles` creati da Amazon EKS sono racchiusi nello spazio dei nomi `kube-system`.

Il comando seguente restituisce tutti i `Roles` Kubernetes creati da Amazon EKS nel tuo cluster.

```
kubectl get roles -n kube-system | grep eks
```

Per visualizzare le specifiche di un `Role`, sostituisci *`eks:k8s-metrics`* nel comando seguente con un `Role` restituito nell'output del comando precedente. L'esempio seguente restituisce la specifica per *`eks:k8s-metrics`* `Role`.

```
kubectl describe role eks:k8s-metrics -n kube-system
```

Di seguito viene riportato un output di esempio:

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
PolicyRule:
  Resources         Non-Resource URLs  Resource Names              Verbs
  ---------         -----------------  --------------              -----
  daemonsets.apps   []                 [aws-node]                  [get]
  deployments.apps  []                 [vpc-resource-controller]  [get]
```

RoleBindings— `RoleBindings` sono limitati a uno Kubernetes spazio dei nomi. Tutti i `RoleBindings` creati da Amazon EKS rientrano nell'ambito dello spazio nomi `kube-system`.

Il comando seguente restituisce tutti i `RoleBindings` Kubernetes creati da Amazon EKS nel tuo cluster.

```
kubectl get rolebindings -n kube-system | grep eks
```

Per visualizzare le specifiche di un `RoleBinding`, sostituisci *`eks:k8s-metrics`* nel comando seguente con un `RoleBinding` restituito nell'output del comando precedente. L'esempio seguente restituisce la specifica per `RoleBinding` *`eks:k8s-metrics`*.

```
kubectl describe rolebinding eks:k8s-metrics -n kube-system
```

Di seguito viene riportato un output di esempio:

```
Name:         eks:k8s-metrics
Labels:       <none>
Annotations:  <none>
Role:
  Kind:  Role
  Name:  eks:k8s-metrics
Subjects:
  Kind  Name             Namespace
  ----  ----             ---------
  User  eks:k8s-metrics
```

Convalida della conformità per Amazon Elastic Kubernetes Service

Per sapere se un Servizio AWS programma rientra nell'ambito di specifici programmi di conformità, consulta Servizi AWS la sezione Scope by Compliance Program Servizi AWS e scegli il programma di conformità che ti interessa. Per informazioni generali, consulta Programmi di AWS conformità Programmi di .

È possibile scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta Scaricamento dei report in AWS Artifact .

La vostra responsabilità di conformità durante l'utilizzo Servizi AWS è determinata dalla sensibilità dei dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. AWS fornisce le seguenti risorse per contribuire alla conformità:

- Guide introduttive su sicurezza e conformità: queste guide all'implementazione illustrano considerazioni sull'architettura e forniscono passaggi per implementare ambienti di base incentrati sulla AWS sicurezza e la conformità.
- Progettazione per la sicurezza e la conformità HIPAA su Amazon Web Services: questo white paper descrive in che modo le aziende possono utilizzare AWS per creare applicazioni idonee all'HIPAA.

> **Note**
>
> Non Servizi AWS tutte sono idonee all'HIPAA. Per ulteriori informazioni, consulta la sezione Riferimenti sui servizi conformi ai requisiti HIPAA.

- AWS Risorse per la per la conformità: questa raccolta di cartelle di lavoro e guide potrebbe essere valida per il tuo settore e la tua località.
- AWS Guide alla conformità dei clienti: comprendi il modello di responsabilità condivisa attraverso la lente della conformità. Le guide riassumono le migliori pratiche per la protezione Servizi AWS e mappano le linee guida per i controlli di sicurezza su più framework (tra cui il National Institute of Standards and Technology (NIST), il Payment Card Industry Security Standards Council (PCI) e l'International Organization for Standardization (ISO)).
- Valutazione delle risorse con regole nella Guida per gli AWS Config sviluppatori: il AWS Config servizio valuta la conformità delle configurazioni delle risorse alle pratiche interne, alle linee guida e alle normative del settore.
- AWS Security Hub— Ciò Servizio AWS fornisce una visione completa dello stato di sicurezza interno. AWS La Centrale di sicurezza utilizza i controlli di sicurezza per valutare le risorse AWS e verificare la conformità agli standard e alle best practice del settore della sicurezza. Per un elenco dei servizi e dei controlli supportati, consulta la pagina Documentazione di riferimento sui controlli della Centrale di sicurezza.
- Amazon GuardDuty: Servizio AWS rileva potenziali minacce ai tuoi carichi di lavoro Account AWS, ai contenitori e ai dati monitorando l'ambiente alla ricerca di attività sospette e dannose. GuardDuty può aiutarti a soddisfare vari requisiti di conformità, come lo standard PCI DSS, soddisfacendo i requisiti di rilevamento delle intrusioni imposti da determinati framework di conformità.
- AWS Audit Manager— Ciò Servizio AWS consente di verificare continuamente l' AWS utilizzo per semplificare la gestione del rischio e la conformità alle normative e agli standard di settore.

Resilienza in Amazon EKS

L'infrastruttura globale di AWS è progettata attorno a Regioni AWS e zone di disponibilità. Regioni AWS fornisce più zone di disponibilità fisicamente separate e isolate che sono connesse tramite reti altamente ridondanti, a bassa latenza e velocità effettiva elevata. Con le zone di disponibilità, è possibile progettare e gestire applicazioni e database che eseguono il failover automatico tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture tradizionali a data center singolo o multiplo.

Amazon EKS esegue ed effettua il dimensionamento di istanze del piano di controllo (control-plane) Kubernetes in molteplici zone di disponibilità AWS per garantire un'elevata disponibilità. Amazon EKS dimensiona automaticamente le istanze del piano di controllo in base al carico, rileva e sostituisce le istanze del piano di controllo non integre e fornisce loro le patch automaticamente. Dopo aver avviato

un aggiornamento della versione, Amazon EKS aggiorna il piano di controllo per conto dell'utente, mantenendone un'elevata disponibilità durante l'aggiornamento.

Questo piano di controllo è costituito da almeno due istanze del server API e tre istanze `etcd` eseguite su tre zone di disponibilità all'interno di una Regione AWS. Amazon EKS:

- Monitora attivamente il carico sulle istanze del piano di controllo e le ridimensiona automaticamente per garantire prestazioni elevate.
- Rileva e sostituisce automaticamente le istanze del piano di controllo non integre, riavviandole nelle zone di disponibilità all'interno della Regione AWS in base alle esigenze.
- Sfrutta l'architettura delle Regioni AWS al fine di mantenere un'elevata disponibilità. Per questo motivo, Amazon EKS è in grado di offrire un SLA per la disponibilità dell'endpoint del server API.

Per ulteriori informazioni sulle Regioni AWS e le zone di disponibilità, consulta Infrastruttura globale di AWS.

Sicurezza dell'infrastruttura in Amazon EKS

In quanto servizio gestito, Amazon Elastic Kubernetes Service è protetto dalla sicurezza di rete globale. AWS Per informazioni sui servizi di AWS sicurezza e su come AWS protegge l'infrastruttura, consulta AWS Cloud Security. Per progettare il tuo AWS ambiente utilizzando le migliori pratiche per la sicurezza dell'infrastruttura, vedi Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Utilizzi chiamate API AWS pubblicate per accedere ad Amazon EKS attraverso la rete. I client devono supportare quanto segue:

- Transport Layer Security (TLS). È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Suite di cifratura con Perfect Forward Secrecy (PFS), ad esempio Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale IAM. O puoi utilizzare AWS Security Token Service (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Quando si crea un cluster Amazon EKS;, specificare anche le sottoreti VPC; che devono essere utilizzate dal cluster. Amazon EKS richiede sottoreti in almeno due zone di disponibilità. È consigliabile un VPC con sottoreti pubbliche e private in modo che Kubernetes possa creare sistemi di bilanciamento del carico pubblici nelle sottoreti pubbliche, che bilancino il carico del traffico ai Pods in esecuzione su nodi che si trovano in sottoreti private.

Per ulteriori informazioni sulle considerazioni per il VPC, consultare Requisiti e considerazioni su VPC e sottoreti di Amazon EKS.

Se crei il tuo VPC e i gruppi di nodi con i AWS CloudFormation modelli forniti nella Guida introduttiva ad Amazon EKS procedura dettagliata, i gruppi di sicurezza del piano di controllo e dei nodi vengono configurati con le nostre impostazioni consigliate.

Per ulteriori informazioni sulle considerazioni per i gruppi di sicurezza, consultare Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS.

Quando si crea un nuovo cluster, Amazon EKS crea un endpoint per il server API Kubernetes gestito utilizzato per comunicare con il cluster (usando strumenti di gestione Kubernetes, ad esempio `kubectl`). Per impostazione predefinita, questo endpoint del server API è pubblico su Internet e l'accesso al server API è protetto utilizzando una combinazione di AWS Identity and Access Management (IAM) e Role Kubernetes Based Access Control (RBAC) nativo.

Puoi abilitare l'accesso privato al server API Kubernetes in modo che tutte le comunicazioni tra i nodi e il server API rimangano all'interno del VPC. È possibile limitare gli indirizzi IP che possono accedere al server API da Internet o disabilitare completamente l'accesso a Internet al server API.

Per ulteriori informazioni sulla modifica dell'accesso all'endpoint del cluster, consultare Modifica dell'accesso all'endpoint del cluster.

È possibile implementare policy di rete Kubernetes con CNI di Amazon VPC o strumenti di terze parti come Project Calico. Per ulteriori informazioni sull'utilizzo della CNI di Amazon VPC per le policy di rete, consulta la pagina Configurazione del cluster per le policy di rete Kubernetes. Project Calico è un progetto open source di terze parti. Per ulteriori informazioni, consulta la documentazione di Project Calico.

Analisi della configurazione e delle vulnerabilità in Amazon EKS

La sicurezza è una considerazione fondamentale per la configurazione e la manutenzione di cluster e applicazioni Kubernetes. Di seguito sono elencate le risorse per analizzare la configurazione di

sicurezza dei cluster EKS, le risorse per verificare le vulnerabilità e le integrazioni con AWS i servizi che possono eseguire l'analisi per te.

Il benchmark Center for Internet Security (CIS) per Amazon EKS

Il Kubernetesbenchmark Center for Internet Security (CIS) fornisce linee guida per le configurazioni di sicurezza di Amazon EKS. Il benchmark:

- È applicabile ai nodi Amazon EC2 (sia gestiti che autogestiti) in cui sei responsabile delle configurazioni della sicurezza per i componenti Kubernetes.
- Fornisce un modo standard approvato dalla community per essere certi di aver configurato in modo sicuro il cluster Kubernetes e i nodi quando si utilizza Amazon EKS.
- È costituito da quattro sezioni: configurazione della registrazione del piano di controllo, configurazioni di sicurezza dei nodi, criteri e servizi gestiti.
- Supporta tutte le versioni di Kubernetes correntemente disponibili in Amazon EKS e può essere eseguito utilizzando kube-bench, uno strumento open source standard per il controllo della configurazione che utilizza il benchmark CIS sui cluster Kubernetes.

Per ulteriori informazioni, consultare Presentazione del benchmark CIS Amazon EKS.

Versioni della piattaforma Amazon EKS

Le versioni della piattaforma Amazon EKS rappresentano le funzionalità del piano di controllo del cluster, inclusi i flag del server Kubernetes API abilitati e l'attuale versione della Kubernetes patch. I nuovi cluster vengono implementati con la versione più recente della piattaforma. Per dettagli, consultare Versioni della piattaforma Amazon EKS.

Puoi aggiornare un cluster Amazon EKS alle versioni più recenti di Kubernetes. Man mano che nuove versioni di Kubernetes diventano disponibili in Amazon EKS, consigliamo di aggiornare tempestivamente i cluster in modo da usare la versione più recente disponibile. Per ulteriori informazioni sulle versioni di Kubernetesin EKS, consulta Versioni Kubernetes di Amazon EKS.

Elenco delle vulnerabilità del sistema operativo

Elenco delle vulnerabilità AL2023

Tieni traccia degli eventi di sicurezza o privacy per Amazon Linux 2023 nell'Amazon Linux Security Center o iscriviti al feed RSS associato. Gli eventi relativi alla sicurezza e alla privacy includono una

panoramica del problema, i pacchetti e le istruzioni per l'aggiornamento delle istanze per correggere il problema.

Elenco delle vulnerabilità di Amazon Linux 2

Tieni traccia degli eventi di sicurezza o privacy per Amazon Linux 2 nell'Amazon Linux Security Center o iscriviti al feed RSS associato. Gli eventi relativi alla sicurezza e alla privacy includono una panoramica del problema, i pacchetti e le istruzioni per l'aggiornamento delle istanze per correggere il problema.

Rilevamento dei nodi con Amazon Inspector

È possibile utilizzare Amazon Inspector per verificare la presenza di accessibilità alla rete non intenzionale dei nodi e le vulnerabilità sulle istanze Amazon EC2.

Rilevamento di cluster e nodi con Amazon GuardDuty

Servizio di rilevamento delle GuardDuty minacce di Amazon che aiuta a proteggere account, contenitori, carichi di lavoro e dati all'interno del tuo AWS ambiente. Tra le altre funzionalità, GuardDuty offre le seguenti due funzionalità che rilevano potenziali minacce ai cluster EKS: EKS Protection e Runtime Monitoring.

Per ulteriori informazioni, consulta Rileva le minacce con Amazon GuardDuty.

Best practice di sicurezza per Amazon EKS

Le best practice di sicurezza per Amazon EKS vengono mantenute su Github: https://aws.github.io/aws-eks-best-practices/security/docs/

Policy di sicurezza pod

Il controller di ammissione della policy di sicurezza di Pod Kubernetes convalida le richieste di creazione e aggiornamento dei Pod rispetto a una serie di regole. Per impostazione predefinita, i cluster Amazon EKS vengono forniti con una policy di sicurezza con autorizzazioni complete senza limitazioni. Per ulteriori informazioni, consulta Policy di sicurezza pod nella documentazione di Kubernetes.

> **Note**
>
> La `PodSecurityPolicy` (PSP) attualmente è obsoleta in Kubernetes versione `1.21` e rimossa in Kubernetes `1.25`. I PSPs verranno sostituiti con PSA (Pod Security Admission), un controller di ammissione integrato che implementa i controlli di sicurezza descritti in PSS (Pod Security Standards). PSA e PSS sono funzioni che entrambe hanno raggiunto lo stato beta e sono abilitate in Amazon EKS per impostazione predefinita. Per gestire la rimozione di PSP in `1.25`, è preferibile implementare PSS in Amazon EKS. Per ulteriori informazioni, consulta Implementazione degli standard di sicurezza Pod in Amazon EKS nel blog AWS.

Policy di sicurezza Pod predefinita di Amazon EKS

I cluster Amazon EKS con Kubernetes versione `1.13` e successive hanno una policy di sicurezza per i Pod predefinita denominata `eks.privileged`. Questa policy non prevede limitazioni sul tipo di Pod che può essere accettato nel sistema, il che equivale all'esecuzione di Kubernetes con il controller `PodSecurityPolicy` disabilitato.

> **Note**
>
> Questa policy è stata creata per mantenere la compatibilità con le versioni precedenti dei cluster che non hanno il controller `PodSecurityPolicy` abilitato. È possibile creare policy più restrittive per il cluster e per i singoli spazi dei nomi e gli account di servizio, quindi eliminare la policy predefinita per abilitare le policy più restrittive.

È possibile visualizzare la policy di default con il comando seguente.

```
kubectl get psp eks.privileged
```

Di seguito viene riportato un output di esempio:

```
NAME              PRIV    CAPS    SELINUX    RUNASUSER    FSGROUP    SUPGROUP
 READONLYROOTFS   VOLUMES
eks.privileged    true    *       RunAsAny   RunAsAny     RunAsAny   RunAsAny    false
      *
```

Per ulteriori dettagli, è possibile descrivere la policy con il comando seguente.

```
kubectl describe psp eks.privileged
```

Di seguito viene riportato un output di esempio:

```
Name:  eks.privileged

Settings:
  Allow Privileged:                       true
  Allow Privilege Escalation:             0xc0004ce5f8
  Default Add Capabilities:               <none>
  Required Drop Capabilities:             <none>
  Allowed Capabilities:                   *
  Allowed Volume Types:                   *
  Allow Host Network:                     true
  Allow Host Ports:                       0-65535
  Allow Host PID:                         true
  Allow Host IPC:                         true
  Read Only Root Filesystem:              false
  SELinux Context Strategy: RunAsAny
    User:                                 <none>
    Role:                                 <none>
    Type:                                 <none>
    Level:                                <none>
  Run As User Strategy: RunAsAny
    Ranges:                               <none>
  FSGroup Strategy: RunAsAny
    Ranges:                               <none>
  Supplemental Groups Strategy: RunAsAny
    Ranges:                               <none>
```

Puoi visualizzare il file YAML completo per la policy di sicurezza Pod `eks.privileged`, il ruolo del cluster e l'associazione del ruolo del cluster in Installazione o ripristinino della policy di sicurezza Pod predefinita.

Eliminazione della policy di sicurezza Pod predefinita di Amazon EKS

Se crei policy più restrittive per i Pods, successivamente potrai eliminare la policy di sicurezza Pod `eks.privileged` di Amazon EKS predefinita per abilitare le policy personalizzate.

> **Important**
>
> Se si utilizza la versione `1.7.0` o successive del plug-in CNI e si assegna una policy di sicurezza del Pod personalizzata all'account del servizio Kubernetes `aws-node` utilizzato per i Pods `aws-node` implementati dal Daemonset, la policy deve avere `NET_ADMIN` nella sezione `allowedCapabilities` e inoltre `hostNetwork: true` e `privileged: true` nella `spec` della policy.

Eliminazione della policy di sicurezza Pod predefinita

1. Creare un file denominato *`privileged-podsecuritypolicy.yaml`* con il contenuto nel file di esempio in Installazione o ripristinino della policy di sicurezza Pod predefinita.
2. Eliminare il file YAML con il comando seguente. In questo modo viene eliminata la policy di sicurezza Pod predefinita, `ClusterRole` e `ClusterRoleBinding` ad esso associato.

```
kubectl delete -f privileged-podsecuritypolicy.yaml
```

Installazione o ripristinino della policy di sicurezza Pod predefinita

Se stai effettuando l'aggiornamento da una versione precedente di Kubernetes o se hai modificato o eliminato la policy di sicurezza Pod `eks.privileged` di Amazon EKS predefinita, potrai ripristinarla completando la seguente procedura.

Installazione o ripristino della policy di sicurezza Pod predefinita

1. Crea un file denominato *`privileged-podsecuritypolicy.yaml`*, con i seguenti contenuti.

```
apiVersion: policy/v1beta1
kind: PodSecurityPolicy
metadata:
  name: eks.privileged
  annotations:
    kubernetes.io/description: 'privileged allows full unrestricted access to
      Pod features, as if the PodSecurityPolicy controller was not enabled.'
    seccomp.security.alpha.kubernetes.io/allowedProfileNames: '*'
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
```

```
spec:
  privileged: true
  allowPrivilegeEscalation: true
  allowedCapabilities:
  - '*'
  volumes:
  - '*'
  hostNetwork: true
  hostPorts:
  - min: 0
    max: 65535
  hostIPC: true
  hostPID: true
  runAsUser:
    rule: 'RunAsAny'
  seLinux:
    rule: 'RunAsAny'
  supplementalGroups:
    rule: 'RunAsAny'
  fsGroup:
    rule: 'RunAsAny'
  readOnlyRootFilesystem: false

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:podsecuritypolicy:privileged
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
rules:
- apiGroups:
  - policy
  resourceNames:
  - eks.privileged
  resources:
  - podsecuritypolicies
  verbs:
  - use

---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
```

```
metadata:
  name: eks:podsecuritypolicy:authenticated
  annotations:
    kubernetes.io/description: 'Allow all authenticated users to create privileged
 Pods.'
  labels:
    kubernetes.io/cluster-service: "true"
    eks.amazonaws.com/component: pod-security-policy
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:podsecuritypolicy:privileged
subjects:
  - kind: Group
    apiGroup: rbac.authorization.k8s.io
    name: system:authenticated
```

2. Applicare il file YAML con il comando seguente.

```
kubectl apply -f privileged-podsecuritypolicy.yaml
```

Domande frequenti sulla policy di sicurezza pod (PSP)

`PodSecurityPolicy` è stata dichiarata obsoleta in Kubernetes`1.21` e rimossa in Kubernetes`1.25`. Se lo utilizzi PodSecurityPolicy nel tuo cluster, devi migrare agli standard di sicurezza Kubernetes Pod integrati (PSS) o a una policy-as-code soluzione prima di aggiornare il cluster alla versione per **`1.25`** evitare interruzioni dei carichi di lavoro. Seleziona una delle domande più frequenti per saperne di più.

Che cos'è una PSP?

PodSecurityPolicyè un controller di ammissione integrato che consente a un amministratore del cluster di controllare gli aspetti delle specifiche sensibili alla sicurezza. Pod Se un Pod soddisfa i requisiti del rispettivo PSP, il Pod viene ammesso al cluster come di consueto. Se un Pod non soddisfa i requisiti PSP, il Pod viene rifiutato e non può essere eseguito.

La rimozione della PSP è specifica di Amazon EKS o viene rimossa in Kubernetes upstream?

Si tratta di una modifica iniziale del progetto Kubernetes e non di una modifica apportata in Amazon EKS. PSP è stata dichiarata obsoleta in Kubernetes `1.21` e rimossa in Kubernetes `1.25`. La community Kubernetes ha identificato gravi problemi di usabilità con la PSP. Questi includono la concessione accidentale di autorizzazioni più ampie del previsto e le difficoltà di ispezione che le PSPs applicano in una determinata situazione. Questi problemi non potevano essere risolti senza apportare modifiche radicali. Questo è il motivo principale per cui la community Kubernetes ha deciso di rimuovere la PSP.

Come posso verificare se sto utilizzando le PSPs nei cluster di Amazon EKS?

Per verificare se stai utilizzando le PSPs nel tuo cluster, esegui il comando seguente:

```
kubectl get psp
```

Per vedere i Pods su cui i PSPs sul cluster influiscono, esegui il comando seguente. Questo comando restituisce il nome del Pod, lo spazio dei nomi e i PSPs:

```
kubectl get pod -A -o jsonpath='{range.items[?(@.metadata.annotations.kubernetes
\.io/psp)]}{.metadata.name}{"\t"}{.metadata.namespace}{"\t"}
{.metadata.annotations.kubernetes\.io/psp}{"\n"}'
```

Cosa posso fare se sto usando le PSPs nel mio cluster Amazon EKS?

Prima di aggiornare il cluster a `1.25`, è necessario migrare le PSPs verso una di queste alternative:

- Kubernetes PSS.

- olicy-as-code Soluzioni P dall'ambiente. Kubernetes

In risposta alla definizione come obsoleta del PSP e alla continua necessità di controllare la sicurezza dei Pod sin dall'inizio, la community Kubernetes ha creato una soluzione integrata con (PSS) e Pod Security Admission (PSA). Il webhook PSA implementa i controlli definiti nel PSS.

Puoi consultare le best practice per la migrazione delle PSPs alla versione integrata della PSS nella Guida alle best practice di EKS. Ti consigliamo anche di consultare il nostro blog sull'implementazione degli standard di sicurezza Pod in Amazon EKS. Ulteriori riferimenti

includono Migrate from PodSecurityPolicy the Built-In PodSecurity Admission Controller e Mapping PodSecurityPolicies to Pod Security Standards.

olicy-as-code Le soluzioni P forniscono barriere per guidare gli utenti del cluster e prevenire comportamenti indesiderati attraverso controlli automatici prescritti. olicy-as-code Le soluzioni P in genere utilizzano i Kubernetes Dynamic Admission Controller per intercettare il flusso di richieste del Kubernetes server API utilizzando una chiamata webhook. olicy-as-code Le soluzioni P modificano e convalidano i payload delle richieste in base a policy scritte e archiviate come codice.

Sono disponibili diverse policy-as-code soluzioni open source per. Kubernetes Per esaminare le migliori pratiche per la migrazione PSPs a una policy-as-code soluzione, consulta la olicy-as-code sezione P della pagina Pod Security su GitHub.

Vedo che una PSP ha chiamato **eks.privileged** nel mio cluster. Di cosa si tratta e cosa posso fare al riguardo?

I cluster Amazon EKS con versione Kubernetes `1.13` e successive hanno una PSP predefinita denominata `eks.privileged`. Questa policy viene creata nei cluster `1.24` e in quelli precedenti. Non viene utilizzata nei cluster `1.25` e in quelli successivi. Amazon EKS esegue automaticamente la migrazione della PSP verso un'applicazione basata su PSS. Non è necessaria nessuna azione da parte tua.

Amazon EKS apporta modifiche alle PSPs già presenti nel mio cluster quando aggiorno il mio cluster alla versione **1.25**?

No. Oltre alla `eks.privileged`, che è una PSP creata da Amazon EKS, non vengono apportate modifiche alle altre PSPs del cluster quando si esegue l'aggiornamento a `1.25`.

Amazon EKS impedirà un aggiornamento del cluster alla versione **1.25** se non ho effettuato la migrazione dalla PSP?

No. Amazon EKS non impedirà l'aggiornamento del cluster alla versione `1.25` se non hai ancora effettuato la migrazione dalla PSP.

Cosa succede se dimentico di migrare PSPs verso PSS/PSA o verso una policy-as-code soluzione prima di aggiornare il cluster alla versione? **1.25** Posso eseguire la migrazione dopo aver aggiornato il cluster?

Quando un cluster che contiene una PSP viene aggiornato a Kubernetes versione `1.25`, il server API non riconosce la risorsa PSP in `1.25`. Ciò potrebbe far sì che i Pods ottengano ambiti di

sicurezza errati. Per un elenco esaustivo delle implicazioni, consulta Migrare da PodSecurityPolicy al PodSecurity Built-In Admission Controller.

In che modo questa modifica influisce sulla sicurezza dei pod per i carichi di lavoro Windows?

Non prevediamo alcun impatto specifico sui carichi di lavoro Windows. PodSecurityContext ha un campo chiamato `windowsOptions` nell'`PodSpec v1`API per WindowsPods. Questo utilizza la PSS in Kubernetes `1.25`. Per ulteriori informazioni e procedure consigliate sull'applicazione di PSS per carichi di lavoro Windows, consulta la Guida alle best practice di EKS e la documentazione di Kubernetes sulle best practice di EKS.

Utilizzo dei segreti di AWS Secrets Manager con Kubernetes

Per mostrare i segreti di Secrets Manager e i parametri dall'archivio parametri come file montati nei Pods Amazon EKS, puoi utilizzare AWS Secrets and Configuration Provider (ASCP) per il driver Secrets Store CSI Kubernetes.

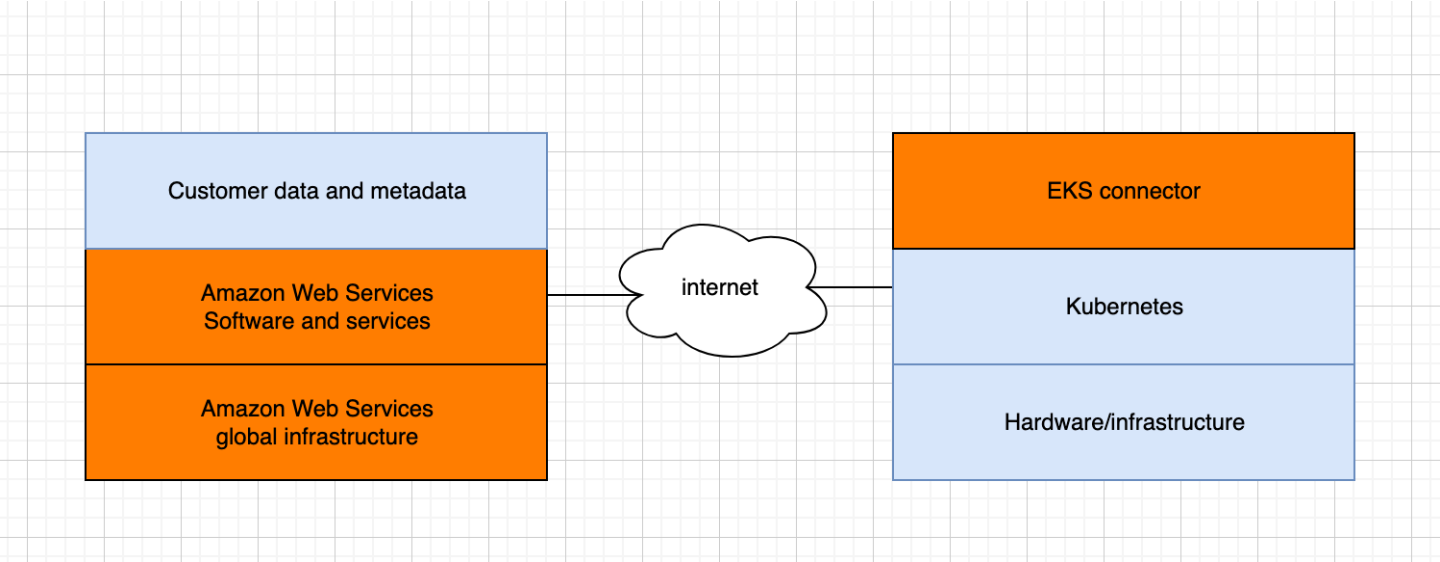
Con ASCP, è possibile archiviare e gestire i segreti in Secrets Manager e recuperarli tramite i carichi di lavoro in esecuzione su Amazon EKS. Puoi utilizzare i ruoli e le policy IAM per limitare l'accesso ai tuoi segreti a specifici Pods Kubernetes in un cluster. ASCP recupera l'identità del Pod e scambia l'identità per un ruolo IAM. ASCP assume il ruolo IAM del Pod e quindi può recuperare i segreti da Secrets Manager autorizzati per tale ruolo.

Se si utilizza la rotazione automatica di Secrets Manager per i propri segreti, è anche possibile utilizzare la funzione di rotation reconciler del Driver CSI di Secrets Store, per assicurarsi di recuperare il segreto più recente da Secrets Manager.

Per ulteriori informazioni, consultare Rotazione dei segreti di Secrets Manager in Amazon EKS nella AWS Guida per l'utente di Secrets Manager.

Considerazioni su Amazon EKS Connector

Amazon EKS Connector è un componente open source che viene eseguito sul cluster Kubernetes. Questo cluster può trovarsi al di fuori dell'ambiente AWS, sollevando alcune considerazioni in materia di responsabilità di sicurezza. Il diagramma seguente illustra tale configurazione. Il colore arancione indica le responsabilità di AWS e il colore blu le responsabilità del cliente:



Questo argomento illustra le differenze nel modello di responsabilità quando il cluster connesso è esterno ad AWS.

Responsabilità di AWS

- Gestione, creazione e implementazione di Amazon EKS Connector, un componente open source che viene eseguito sul cluster Kubernetes di un cliente e comunica con AWS.
- Mantenimento della sicurezza delle comunicazioni a livello di trasporto e di applicazione tra il cluster Kubernetes connesso e i servizi AWS.

Responsabilità del cliente

- Sicurezza specifica del cluster Kubernetes, in particolare seguendo le linee guida riportate di seguito:
  - I segreti di Kubernetes devono essere adeguatamente crittografati e protetti.
  - L'accesso allo spazio dei nomi `eks-connector` deve essere bloccato.
- Configurazione delle autorizzazioni relative al controllo degli accessi basato sul ruolo (RBAC) per gestire l'accesso del principale IAM da AWS. Per istruzioni, consultare Concessione dell'accesso a un principale IAM per la visualizzazione delle risorse Kubernetes in un cluster.
- Installazione e aggiornamento di Amazon EKS Connector.
- Manutenzione dell'hardware, del software e dell'infrastruttura che supportano il cluster Kubernetescollegato.

- Protezione degli account AWS (ad esempio, tramite la salvaguardia delle credenziali dell'utente root).

Visualizzazione delle risorse Kubernetes

Puoi visualizzare le risorse Kubernetes implementate nel cluster utilizzando la AWS Management Console. Non puoi visualizzare Kubernetes le risorse con AWS CLI o `eksctl`. Per visualizzare le risorse Kubernetes con uno strumento a riga di comando, utilizza `kubectl`.

Prerequisito

Per visualizzare la scheda Risorse e la sezione Nodi nella scheda Compute AWS Management Console, il principale IAM che stai utilizzando deve disporre di IAM e Kubernetes autorizzazioni specifici. Per ulteriori informazioni, consulta Autorizzazioni richieste.

Per visualizzare le Kubernetes risorse con AWS Management Console

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Nell'elenco Clusters (Cluster), seleziona il cluster contenente le risorse Kubernetes da visualizzare.
3. Selezionare la scheda Risorse.
4. Seleziona un gruppo Resource type (Tipi di risorse) per il quale desideri visualizzare le risorse, ad esempio Workloads (Carichi di lavoro). Viene visualizzato un elenco dei tipi di risorse in tale gruppo.
5. Seleziona un tipo di risorsa, ad esempio Deployments (Implementazioni), nel gruppo Workloads (Carichi di lavoro). Viene visualizzata una descrizione del tipo di risorsa, un collegamento alla documentazione Kubernetes per ulteriori informazioni sul tipo di risorsa e un elenco di risorse simili implementate nel cluster. Se l'elenco è vuoto, nel cluster non sono presenti risorse appartenenti a quel tipo specifico.
6. Per visualizzare ulteriori informazioni su una risorsa, selezionarla. Prova gli esempi seguenti:

    - Seleziona il gruppo Workloads (Carichi di lavoro), scegli il tipo di risorsa Deployments (Implementazioni), quindi seleziona la risorsa coredns. Per impostazione predefinita, la selezione di una risorsa avviene in una Visualizzazione strutturata. Per alcuni tipi di risorse, nella Visualizzazione strutturata è presente una sezione Pod, con un elenco dei Pods gestiti dal carico di lavoro. Puoi selezionare qualsiasi Pod presente in elenco per visualizzarne le informazioni. Non tutti i tipi di risorse visualizzano le informazioni nella Visualizzazione strutturata. Selezionando Raw view (Visualizzazione non elaborata) nell'angolo in alto a destra della pagina della risorsa, viene visualizzata la risposta JSON completa dall'API Kubernetes per la risorsa.

- Seleziona il gruppo Cluster, quindi il tipo di risorsa Nodes (Nodi). Viene visualizzato un elenco di tutti i nodi del cluster. I nodi possono appartenere a qualsiasi tipo di nodo di Amazon EKS. Si tratta dello stesso elenco visualizzato nella sezione Nodes (Nodi) quando si seleziona la scheda Compute (Calcolo) del cluster. Seleziona una risorsa di nodo dall'elenco. Nella Vista strutturata è presente anche la sezione Pod contenente tutti i Pods in esecuzione sul nodo.

Autorizzazioni richieste

Per visualizzare la scheda Risorse e la sezione Nodi nella scheda Compute AWS Management Console, il principale IAM che stai utilizzando deve disporre di IAM e Kubernetes autorizzazioni minimi specifici. Completa i passaggi seguenti per assegnare le autorizzazioni richieste ai principali IAM.

1. Assicurati che `eks:AccessKubernetesApi` e le altre autorizzazioni IAM necessarie per visualizzare le risorse Kubernetes siano assegnati al principale IAM che stai utilizzando. Per ulteriori informazioni su come modificare le autorizzazioni per un principale IAM, consulta Controllo dell'accesso per i principali IAM nella Guida per l'utente di IAM. Per ulteriori informazioni su come modificare le autorizzazioni per un ruolo, consulta Modifica di una policy di autorizzazioni del ruolo (console) nella Guida per l'utente IAM.

   La seguente policy di esempio include le autorizzazioni necessarie affinché un principale possa visualizzare le risorse Kubernetes per tutti i cluster dell'account. Sostituisci *111122223333* con il tuo ID account AWS .

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "eks:ListFargateProfiles",
                   "eks:DescribeNodegroup",
                   "eks:ListNodegroups",
                   "eks:ListUpdates",
                   "eks:AccessKubernetesApi",
                   "eks:ListAddons",
                   "eks:DescribeCluster",
                   "eks:DescribeAddonVersions",
                   "eks:ListClusters",
   ```

```
                "eks:ListIdentityProviderConfigs",
                "iam:ListRoles"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ssm:GetParameter",
            "Resource": "arn:aws:ssm:*:111122223333:parameter/*"
        }
    ]
}
```

Per visualizzare i nodi in cluster connessi, il ruolo IAM del connettore Amazon EKS dovrebbe essere in grado di rappresentare il principale nel cluster. Ciò consente a Amazon EKS Connector di mappare il principale a un utente Kubernetes.

2. Crea un `rolebinding` o `clusterrolebinding` Kubernetes associato a un `role` o `clusterrole` Kubernetes che abbia le autorizzazioni necessarie per visualizzare le risorse Kubernetes. Per ulteriori informazioni sui ruoli Kubernetes e sulle relative associazioni, consulta Uso dell'autorizzazione RBAC nella documentazione Kubernetes. Puoi applicare al cluster uno dei seguenti manifesti che consentono di creare un `role` e `rolebinding` o un `clusterrole` e `clusterrolebinding` con le autorizzazioni Kubernetes necessarie:

   Visualizzazione delle risorse Kubernetes in tutti gli spazi dei nomi

   Il nome del gruppo nel file è `eks-console-dashboard-full-access-group`. Applica il manifesto al cluster con il comando seguente:

   ```
   kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
   ```

   Visualizzazione delle risorse Kubernetes in uno spazio dei nomi specifico

   Lo spazio dei nomi in questo file è `default`. Il nome del gruppo nel file è `eks-console-dashboard-restricted-access-group`. Applica il manifesto al cluster con il comando seguente:

   ```
   kubectl apply -f https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
   ```

Per modificare il nome del gruppo Kubernetes, lo spazio dei nomi, le autorizzazioni o qualsiasi altra configurazione, scarica il file e modificalo prima di applicarlo al cluster:

1. Scarica il file tramite uno dei comandi seguenti:

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-full-access.yaml
```

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/docs/eks-console-restricted-access.yaml
```

2. Modifica il file in base alle esigenze.
3. Applica il manifesto al cluster con uno dei comandi seguenti:

```
kubectl apply -f eks-console-full-access.yaml
```

```
kubectl apply -f eks-console-restricted-access.yaml
```

3. Mappa il principale IAM all'utente Kubernetes o al gruppo in `aws-auth ConfigMap`. Per aggiornare `ConfigMap` puoi usare uno strumento adeguato, ad esempio `eksctl`, oppure puoi eseguire l'aggiornamento in modo manuale tramite modifica.

> **Important**
>
> Ti consigliamo di utilizzare `eksctl`, o uno strumento simile, per modificare `ConfigMap`. Per informazioni su altri strumenti che è possibile utilizzare, consulta Utilizzo degli strumenti per apportare modifiche alla `aws-authConfigMap` nelle guide alle best practice di Amazon EKS. Una formattazione impropria di `aws-auth ConfigMap` può causare la perdita dell'accesso al cluster.

eksctl

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

1. Visualizza le mappature correnti in `ConfigMap`. Sostituisci *`my-cluster`* con il nome del cluster. *`region-code`*Sostituiscilo con Regione AWS quello in cui si trova il cluster.

   ```
   eksctl get iamidentitymapping --cluster my-cluster --region=region-code
   ```

   Di seguito viene riportato un output di esempio:

   ```
   ARN
                     USERNAME                                GROUPS
               ACCOUNT
   arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
   NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
    system:bootstrappers,system:nodes
   ```

2. Aggiungi una mappatura per un ruolo. Per questo esempio si presuppone che le autorizzazioni IAM siano state collegate nel primo passaggio a un ruolo denominato *`my-console-viewer-role`*. Sostituisci *`111122223333`* con l'ID del tuo account.

   ```
   eksctl create iamidentitymapping \
       --cluster my-cluster \
       --region=region-code \
       --arn arn:aws:iam::111122223333:role/my-console-viewer-role \
       --group eks-console-dashboard-full-access-group \
       --no-duplicate-arns
   ```

   > **Important**
   >
   > L'ARN del ruolo non può includere un percorso, ad esempio `role/my-team/developers/my-role`. Il formato dell'ARN deve essere `arn:aws:iam::111122223333:role/my-role`. In questo esempio, `my-team/developers/` deve essere rimosso.

   Di seguito viene riportato un output di esempio.

```
[...]
2022-05-09 14:51:20 [#]  adding identity "arn:aws:iam::111122223333:role/my-
console-viewer-role" to auth ConfigMap
```

3. Aggiungi una mappatura per un utente. Le best practice IAM consigliano di concedere le autorizzazioni ai ruoli anziché agli utenti. Per questo esempio si presuppone che le autorizzazioni IAM siano state collegate a un utente denominato *my-user* nel primo passaggio. Sostituisci *111122223333* con l'ID del tuo account.

```
eksctl create iamidentitymapping \
    --cluster my-cluster \
    --region=region-code \
    --arn arn:aws:iam::111122223333:user/my-user \
    --group eks-console-dashboard-restricted-access-group \
    --no-duplicate-arns
```

   Di seguito viene riportato un output di esempio:

```
[...]
2022-05-09 14:53:48 [#]  adding identity "arn:aws:iam::111122223333:user/my-
user" to auth ConfigMap
```

4. Visualizza nuovamente le mappature nella `ConfigMap`.

```
eksctl get iamidentitymapping --cluster my-cluster --region=region-code
```

   Di seguito viene riportato un output di esempio:

```
ARN
                  USERNAME                                GROUPS
                    ACCOUNT
arn:aws:iam::111122223333:role/eksctl-my-cluster-my-nodegroup-
NodeInstanceRole-1XLS7754U3ZPA    system:node:{{EC2PrivateDNSName}}
 system:bootstrappers,system:nodes
arn:aws:iam::111122223333:role/my-console-viewer-role
                                                          eks-console-
dashboard-full-access-group
arn:aws:iam::111122223333:user/my-user
                                                          eks-console-
dashboard-restricted-access-group
```

Edit ConfigMap manually

Per ulteriori informazioni sull'aggiunta degli utenti o dei ruoli a `aws-auth ConfigMap`, consulta Aggiunta di principali IAM al cluster Amazon EKS.

1. Apri `aws-auth ConfigMap` per la modifica.

```
kubectl edit -n kube-system configmap/aws-auth
```

2. Aggiungi le mappature a `aws-auth ConfigMap` senza sostituire quelle esistenti. L'esempio seguente mostra l'aggiunta di mappature tra principali IAM con le autorizzazioni aggiunte nel primo passaggio e i gruppi Kubernetes creati nel passaggio precedente:
   - Il ruolo *my-console-viewer-role* e il `eks-console-dashboard-full-access-group`.
   - L'utente *my-user* e il `eks-console-dashboard-restricted-access-group`.

   Per questi esempi si presuppone che le autorizzazioni IAM siano state collegate nel primo passaggio a un ruolo denominato *my-console-viewer-role* e a un utente denominato *my-user*. *111122223333*Sostituiscilo con l'ID AWS del tuo account.

```
apiVersion: v1
data:
mapRoles: |
  - groups:
    - eks-console-dashboard-full-access-group
    rolearn: arn:aws:iam::111122223333:role/my-console-viewer-role
    username: my-console-viewer-role
mapUsers: |
  - groups:
    - eks-console-dashboard-restricted-access-group
    userarn: arn:aws:iam::111122223333:user/my-user
    username: my-user
```

> **Important**
>
> L'ARN del ruolo non può includere un percorso, ad esempio `role/my-team/developers/my-console-viewer-role`. Il formato dell'ARN deve essere

`arn:aws:iam::111122223333:role/my-console-viewer-role`. In questo esempio, `my-team/developers/` deve essere rimosso.

3. Salva il file ed esci dall'editor di testo.

Osservabilità in Amazon EKS

Amazon EKS mette a disposizione una serie di strumenti di monitoraggio o registrazione per l'osservazione dei dati. I dati di log di Amazon EKS possono essere trasmessi in Servizi AWS streaming a strumenti partner per l'analisi dei dati. Sono disponibili molti servizi AWS Management Console che forniscono dati per la risoluzione dei problemi di Amazon EKS. Puoi anche utilizzare una soluzione open source AWS supportata per monitorare l'infrastruttura Amazon EKS.

Dopo aver selezionato Cluster nel riquadro di navigazione sinistro della console Amazon EKS, puoi visualizzare l'integrità e i dettagli del cluster selezionandone il nome. Per visualizzare i dettagli delle risorse Kubernetes esistenti implementate nel cluster, consulta Visualizzazione delle risorse Kubernetes.

Il monitoraggio è una parte importante per mantenere l'affidabilità, la disponibilità e le prestazioni di Amazon EKS e delle tue AWS soluzioni. Ti consigliamo di raccogliere dati di monitoraggio da tutte le parti della AWS soluzione. per consentire un debug più facile di eventuali errori in più punti. Prima di iniziare il monitoraggio di Amazon EKS, è opportuno creare un piano di monitoraggio che tenga conto dei seguenti aspetti.

- Quali sono gli obiettivi? Sono necessarie notifiche in tempo reale se i cluster si ridimensionano notevolmente?
- Di quali risorse si intende eseguire il monitoraggio?
- Con quale frequenza sarà eseguito il monitoraggio di queste risorse? L'azienda vuole rispondere rapidamente ai rischi?
- Quali strumenti verranno utilizzati? Se l'hai già eseguita AWS Fargate come parte del lancio, puoi utilizzare il router di registro integrato.
- Chi eseguirà le attività di monitoraggio?
- Chi deve ricevere una notifica quando si verifica un problema?

Registrazione e monitoraggio su Amazon EKS

Amazon EKS offre strumenti integrati per la registrazione e il monitoraggio. I log del piano di controllo (control-plane) registrano tutte le chiamate dell'API ai tuoi cluster, le informazioni di controllo che acquisiscono quali utenti hanno eseguito determinate azioni sui tuoi cluster, e informazioni basate sui ruoli. Per ulteriori informazioni, consulta la sezione Registrazione e monitoraggio su Amazon EKS nella Guida prescrittiva di AWS .

La registrazione del piano di controllo di Amazon EKS fornisce log di audit e diagnostica direttamente dal piano di controllo di Amazon CloudWatch EKS ai log del tuo account. Questi log consentono di semplificare la protezione e l'esecuzione dei cluster. Puoi selezionare i tipi di log esatti di cui hai bisogno e i log vengono inviati come flussi di log a un gruppo per ogni cluster Amazon EKS in cui opera. CloudWatch Per ulteriori informazioni, consulta Logging del piano di controllo di Amazon EKS.

> **Note**
>
> Quando controlli i log dell'autenticatore di Amazon EKS su Amazon CloudWatch, vengono visualizzate le voci che contengono testo simile al seguente testo di esempio.
>
> ```
> level=info msg="mapping IAM role" groups="[]"
>  role="arn:aws:iam::111122223333:role/XXXXXXXXXXXXXXXXX-
> NodeManagerRole-XXXXXXXX" username="eks:node-manager"
> ```
>
> Sono previste voci che contengono questo testo. Lo `username` è un ruolo di servizio Amazon EKS interno che esegue operazioni specifiche per i gruppi di nodi gestiti e nodi Fargate. Per la registrazione personalizzabile di basso livello, consulta Registrazione Kubernetes.

Amazon EKS è integrato con AWS CloudTrail, un servizio che fornisce un registro delle azioni intraprese da un utente, un ruolo o un AWS servizio in Amazon EKS. CloudTrail acquisisce tutte le chiamate API per Amazon EKS come eventi. Le chiamate acquisite includono le chiamate dalla console Amazon EKS e le chiamate di codice alle operazioni API Amazon EKS. Per ulteriori informazioni, consulta Registrazione delle chiamate API di Amazon EKS con AWS CloudTrail.

Il server API Kubernetes espone una serie di parametri che sono utili per il monitoraggio e l'analisi dei dati. Per ulteriori informazioni, consulta Parametri di Prometheus.

Fluent BitPer configurare Amazon CloudWatch i log personalizzati, consulta Configurazione Fluent Bit nella Amazon CloudWatch User Guide.

Strumenti di registrazione e monitoraggio di Amazon EKS

Amazon Web Services offre vari strumenti che permettono di monitorare Amazon EKS. Alcuni di questi strumenti possono essere configurati in modo che eseguano automaticamente il monitoraggio, mentre altri richiedono l'intervento manuale. Ti consigliamo di automatizzare il più possibile i processi di monitoraggio nella misura in cui l'ambiente e il set di strumenti esistenti lo consentono.

Strumenti di registrazione

Aree	Strumento	Descrizione	Installazione
Applicazioni	Amazon CloudWatch Container Insights	Raccoglie, aggrega e riepiloga i parametri e i registri di applicazioni e microservizi containerizzati.	Procedura di configurazione
Piano di controllo (control-plane)	AWS CloudTrail	Registra le chiamate API da parte di un utente, un ruolo o un servizio.	Procedura di configurazione
Diverse aree per le AWS Fargate istanze	AWS Fargate router di registro	AWS Fargate Ad esempio, trasmette i log ai AWS servizi o agli strumenti dei partner. Usa AWS per Fluent Bit. I log possono essere trasmessi in streaming ad altri strumenti o a quelli dei partner. Servizi AWS	Procedura di configurazione

Strumenti di monitoraggio

Aree	Strumento	Descrizione	Installazione
Applicazioni	CloudWatch Container Insights	CloudWatch Container Insights raccoglie, aggrega e riepiloga metriche e log delle applicazioni e dei microservizi containerizzati.	Procedura di configurazione
Applicazioni	AWS Distro per OpenTelemetry (ADOT)	Raccoglie e invia metriche, dati di traccia e metadati correlati a servizi o partner di monitoraggio. AWS Può essere configurato tramite Container Insights. CloudWatch	Procedura di configurazione
Applicazioni	Amazon DevOps Guru	Rileva le prestazioni operative e la disponibilità a livello di nodo.	Procedura di configurazione
Applicazioni	AWS X-Ray	Riceve i dati di traccia dell'applicazione. Questi	Procedura di configurazione

Aree	Strumento	Descrizione	Installazione
		dati di traccia includono le richieste in entrata e in uscita e i metadati relativi a tali richieste . L'impleme ntazione per Amazon EKS richiede il component e aggiuntivo OpenTelemetry.	
Applicazioni	Operatore di CloudWatch osservabilità di Amazon	Amazon CloudWatc h Observabi lity Operator raccoglie metriche, log e dati di traccia. Li invia ad Amazon CloudWatch e AWS X-Ray.	Procedura di configurazione
Piano di controllo (control-plane)	Prometheus	CloudWatc h Le tariffe di inserimento dei log, archiviaz ione e scansione dei dati si applicano ai log del piano di controllo abilitati.	Procedura di configurazione

Parametri di Prometheus

Prometheus è un database di monitoraggio e di serie temporali che esegue lo scraping degli endpoint. Offre la possibilità di eseguire query, aggregare e archiviare i dati raccolti. È possibile utilizzarlo anche per gli avvisi e l'aggregazione degli avvisi. Questo argomento spiega come configurare Prometheus come un'opzione gestita oppure open source. Il monitoraggio dei parametri del piano di controllo di Amazon EKS è un caso d'uso comune.

Amazon Managed Service for Prometheus è un servizio di monitoraggio e avviso compatibile con Prometheus che semplifica il monitoraggio di infrastrutture e applicazioni containerizzate su larga scala. È un servizio completamente gestito che dimensiona automaticamente l'importazione, l'archiviazione, le query e gli avvisi dei parametri. Si integra inoltre con i servizi AWS di sicurezza per consentire un accesso rapido e sicuro ai dati. È possibile utilizzare il linguaggio di query open source ProMQL per fare una query e creare avvisi relativi ai parametri.

Per ulteriori informazioni su come utilizzare i parametri Prometheus dopo averli attivati, consulta la Guida per l'utente di Amazon Managed Service for Prometheus.

Attivazione dei parametri Prometheus durante la creazione di un cluster

> **Important**
>
> Le risorse di Amazon Managed Service for Prometheus non rientrano nel ciclo di vita del cluster e devono essere gestite indipendentemente dal cluster. Quando elimini il cluster, assicurati di eliminare anche tutti gli scraper applicabili per bloccare i costi applicabili. Per ulteriori informazioni, consulta Trova ed elimina gli scraper nella Guida per l'utente di Amazon Managed Service for Prometheus.

Quando si crea un nuovo cluster, è possibile attivare l'opzione per inviare i parametri a Prometheus. Nella AWS Management Console, questa opzione si trova nella fase Configura l'osservabilità della creazione di un nuovo cluster. Per ulteriori informazioni, consulta Creazione di un cluster Amazon EKS.

Prometheus rileva e raccoglie i parametri dal cluster tramite un modello basato su pull chiamato scraping. Gli scraper sono configurati per raccogliere dati dall'infrastruttura del cluster e dalle applicazioni containerizzate.

Quando attivi l'opzione di invio dei parametri Prometheus, Amazon Managed Service for Prometheus fornisce uno scraper agentless completamente gestito. Utilizza le seguenti opzioni di configurazione avanzata per personalizzare lo scraper predefinito in base alle esigenze.

Alias scraper

(Facoltativo) Inserisci un alias univoco per lo scraper.

Destinazione

Scegli un workspace Amazon Managed Service for Prometheus. Un workspace è uno spazio logico dedicato all'archiviazione e alle query dei parametri Prometheus. Con questo workspace, sarai in grado di visualizzare i parametri Prometheus di tutti gli account che vi hanno accesso. L'opzione Crea nuovo workspace consente ad Amazon EKS di creare un workspace per tuo conto utilizzando l'alias del workspace che fornisci. Con l'opzione Seleziona workspace esistente, puoi selezionare un workspace esistente da un elenco a discesa. Per ulteriori informazioni sui workspace, consulta Gestione di workspace nella Guida per l'utente di Amazon Managed Service for Prometheus.

Accesso al servizio

Questa sezione riassume le autorizzazioni concesse per l'invio di parametri Prometheus:

- Consentire ad Amazon Managed Service for Prometheus di descrivere il cluster Amazon EKS sottoposto a scraping
- Consentire la scrittura remota nel workspace Amazon Managed Prometheus

Se `AmazonManagedScraperRole` esiste già, viene utilizzato dallo scraper. Scegli il link `AmazonManagedScraperRole` per visualizzare i dettagli dell'autorizzazione. Se `AmazonManagedScraperRole` non esiste già, scegli il link Visualizza dettagli autorizzazione per vedere le autorizzazioni specifiche che concedi inviando i parametri Prometheus.

Sottoreti

Visualizza le sottoreti che vengono ereditate dallo scraper. Se è necessario modificarle, torna al passaggio di creazione del cluster Specificare la rete.

Gruppi di sicurezza

Visualizza i gruppi di sicurezza che verranno ereditati dallo scraper. Se è necessario modificarle, torna al passaggio di creazione del cluster Specificare la rete.

Configurazione dello scraper

Modifica la configurazione dello scraper in formato YAML secondo le esigenze. A tale scopo, utilizza il modulo o carica un file YAML sostitutivo. Per ulteriori informazioni, consulta Configurazione dello scraper nella Guida per l'utente di Amazon Managed Service for Prometheus.

Amazon Managed Service for Prometheus si riferisce allo scraper agentless creato insieme al cluster come raccoglitore gestito da AWS . Per ulteriori informazioni sui raccoglitori AWS gestiti, consulta AWS Managed Collector nella Guida per l'utente di Amazon Managed Service for Prometheus.

> **Important**
>
> È necessario configurare `aws-auth ConfigMap` per fornire allo scraper le autorizzazioni in-cluster. Per ulteriori informazioni, consulta Configurazione del cluster Amazon EKS nella Guida per l'utente di Amazon Managed Service for Prometheus.

Visualizzazione dei dettagli dello scraper Prometheus

Dopo aver creato un cluster con l'opzione Parametri Prometheus attivata, puoi visualizzare i dettagli dello scraper Prometheus. Quando visualizzi il cluster in AWS Management Console, scegli la scheda Osservabilità. Una tabella mostra un elenco di scraper per il cluster, incluse informazioni come l'ID, l'alias, lo stato e la data di creazione dello scraper.

Per visualizzare ulteriori dettagli sullo scraper, scegli un collegamento relativo all'ID dello scraper. Ad esempio, puoi visualizzare la configurazione dello scraper, l'ARN (Amazon Resource Name), l'URL di scrittura remota e le informazioni sulla rete. Puoi utilizzare l'ID scraper come input per operazioni dell'API di Amazon Managed Service for Prometheus come `DescribeScraper` e `DeleteScraper`. Puoi anche utilizzare l'API per creare altri scraper.

Per ulteriori informazioni sull'uso dell'API Prometheus, consulta la Documentazione di riferimento delle API di Amazon Managed Service for Prometheus.

Implementazione di Prometheus utilizzando Helm

In alternativa, puoi eseguire l'implementazione di Prometheus nel tuo cluster con Helm V3. Se hai già installato Helm, puoi controllarne la versione con il comando `helm version`. Helm è un gestore di

pacchetti per cluster Kubernetes. Per ulteriori informazioni su Helm e su come installarlo, consulta Utilizzo di Helm con Amazon EKS.

Dopo aver configurato Helm per il tuo cluster Amazon EKS, puoi utilizzarlo per implementare Prometheus con i passaggi seguenti.

Per implementare Prometheus utilizzando Helm

1. Crea uno spazio dei nomi Prometheus.

```
kubectl create namespace prometheus
```

2. Aggiungere il repository del grafico `prometheus-community`.

```
helm repo add prometheus-community https://prometheus-community.github.io/helm-charts
```

3. Implementa Prometheus.

```
helm upgrade -i prometheus prometheus-community/prometheus \
    --namespace prometheus \
    --set alertmanager.persistence.storageClass="gp2" \
    --set server.persistentVolume.storageClass="gp2"
```

> **Note**
>
> Se si riceve l'errore `Error: failed to download "stable/prometheus" (hint: running `helm repo update` may help)` durante l'esecuzione di questo comando, eseguire `helm repo update prometheus-community`, quindi provare a eseguire nuovamente il comando della fase 2.
> Se si riceve l'errore `Error: rendered manifests contain a resource that already exists`, eseguire `helm uninstall` *`your-release-name`* `-n` *`namespace`*, quindi provare a eseguire nuovamente il comando della fase 3.

4. Verifica che tutti i Pods nello spazio dei nomi `prometheus` si trovino nello stato `READY`.

```
kubectl get pods -n prometheus
```

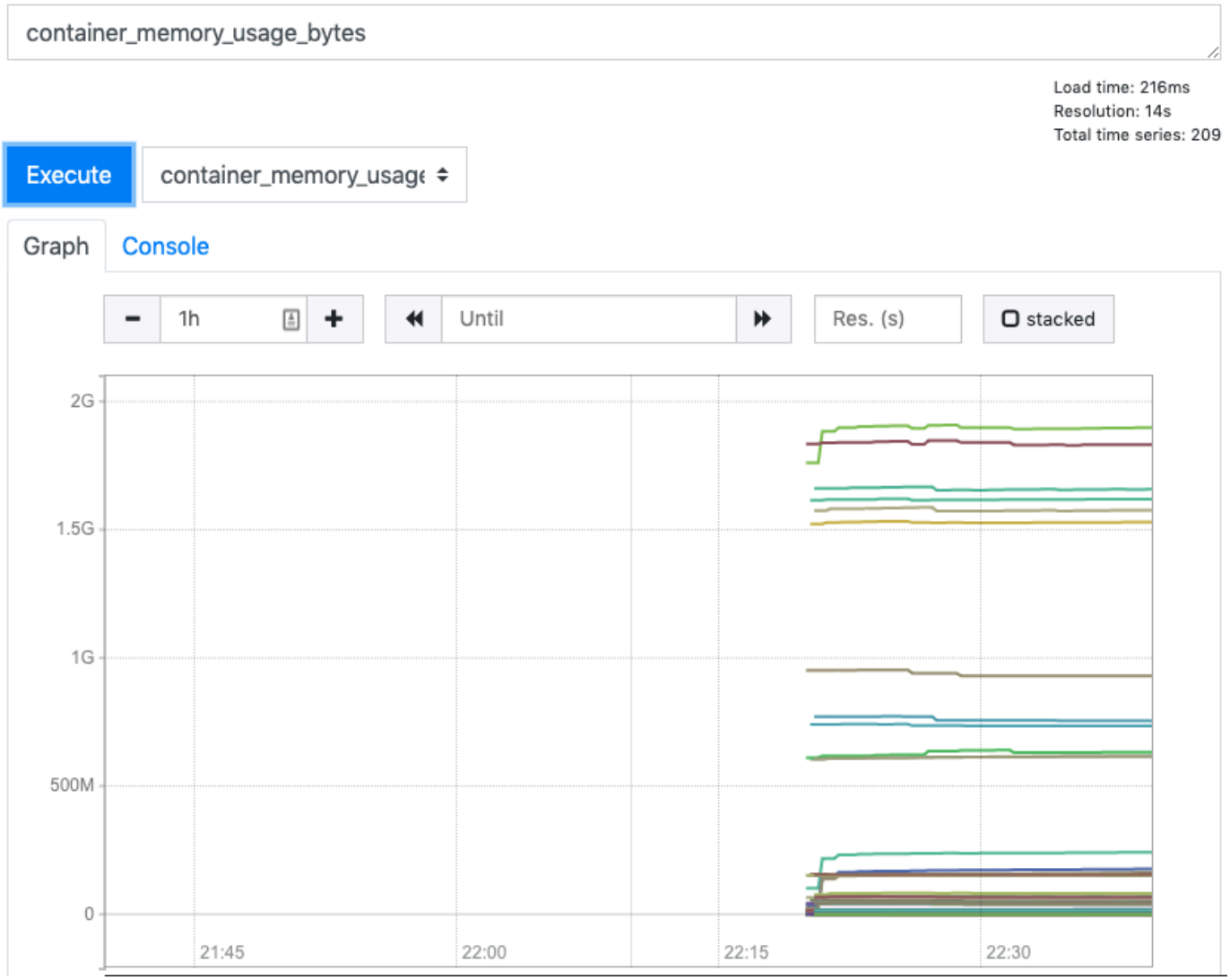
Di seguito viene riportato un output di esempio:

```
NAME                                             READY   STATUS    RESTARTS   AGE
prometheus-alertmanager-59b4c8c744-r7bgp         1/2     Running   0          48s
prometheus-kube-state-metrics-7cfd87cf99-jkz2f   1/1     Running   0          48s
prometheus-node-exporter-jcjqz                   1/1     Running   0          48s
prometheus-node-exporter-jxv2h                   1/1     Running   0          48s
prometheus-node-exporter-vbdks                   1/1     Running   0          48s
prometheus-pushgateway-76c444b68c-82tnw          1/1     Running   0          48s
prometheus-server-775957f748-mmht9               1/2     Running   0          48s
```

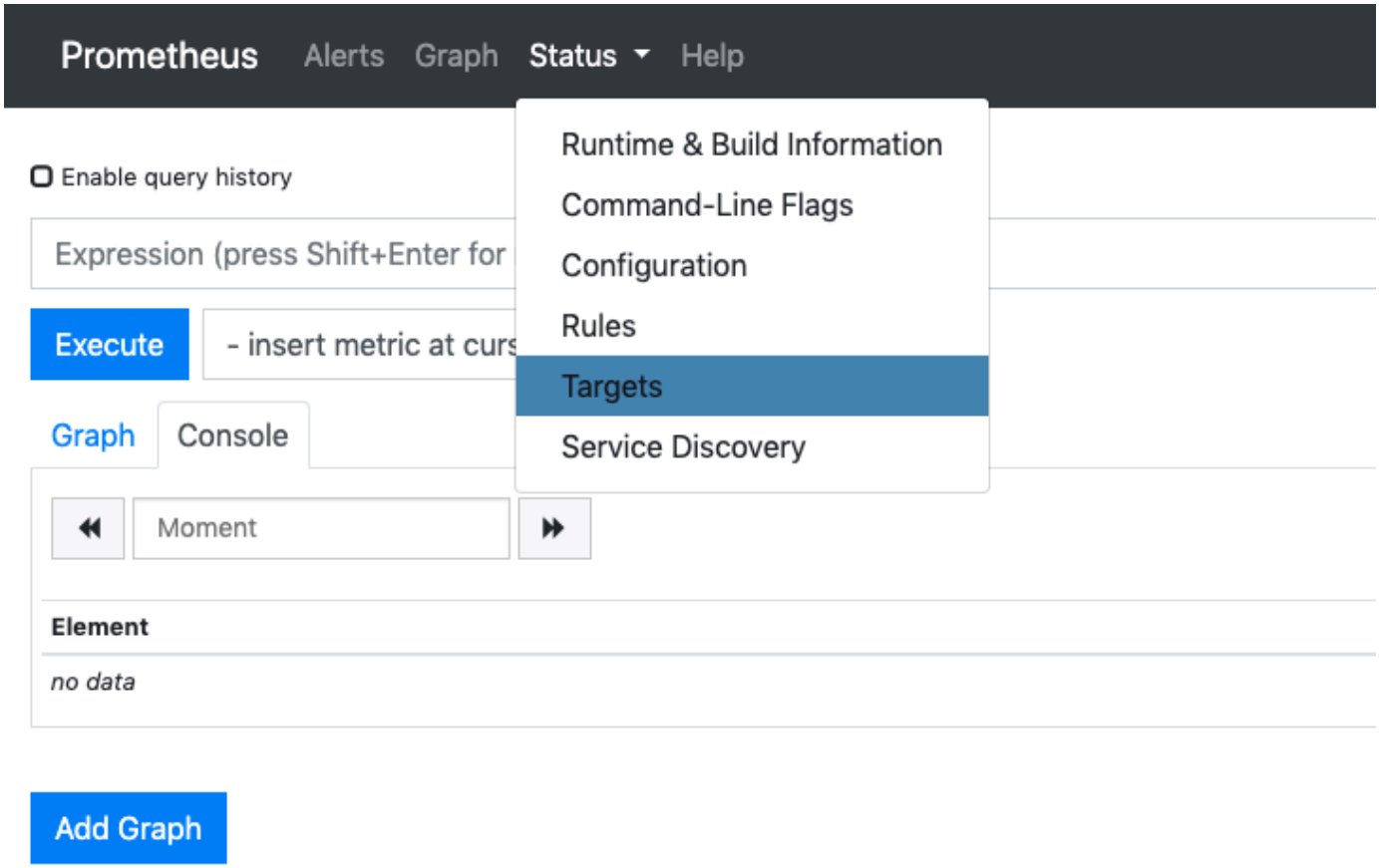
5. Utilizza `kubectl` per l'inoltro della porta della console Prometheus alla tua macchina locale.

```
kubectl --namespace=prometheus port-forward deploy/prometheus-server 9090
```

6. Indirizza un browser web a `http://localhost:9090` per visualizzare la console Prometheus.
7. Scegliere un parametro dal menu - insert metric at cursor, quindi selezionare Execute (Esegui). Scegliere la scheda Graph (Grafico) per visualizzare il parametro nel tempo. L'immagine che segue mostra `container_memory_usage_bytes` nel tempo.



8. Dalla barra di navigazione superiore, scegliere Status (Stato), quindi Targets (Destinazioni).



Vengono visualizzati tutti gli endpoint di Kubernetes connessi a Prometheus tramite il rilevamento servizi.

Visualizzazione dei parametri non elaborati del piano di controllo

In alternativa all'implementazione di Prometheus, il server API Kubernetes espone una serie di parametri rappresentati in un formato Prometheus. Questi parametri sono utili per il monitoraggio e l'analisi. Vengono esposti internamente tramite un endpoint dei parametri che fa riferimento all'API HTTP `/metrics`. Come altri endpoint, questo è esposto sul piano di controllo Amazon EKS. Questo endpoint è utile principalmente per esaminare un parametro specifico. Per analizzare i parametri nel tempo, consigliamo di implementare Prometheus.

Per visualizzare parametri non elaborati, utilizza `kubectl` con il flag `--raw`. Questo comando consente di inoltrare qualsiasi percorso HTTP e restituisce la risposta non elaborata.

```
kubectl get --raw /metrics
```

Di seguito viene riportato un output di esempio:

```
[...]
# HELP rest_client_requests_total Number of HTTP requests, partitioned by status code,
 method, and host.
# TYPE rest_client_requests_total counter
rest_client_requests_total{code="200",host="127.0.0.1:21362",method="POST"} 4994
rest_client_requests_total{code="200",host="127.0.0.1:443",method="DELETE"} 1
rest_client_requests_total{code="200",host="127.0.0.1:443",method="GET"} 1.326086e+06
rest_client_requests_total{code="200",host="127.0.0.1:443",method="PUT"} 862173
rest_client_requests_total{code="404",host="127.0.0.1:443",method="GET"} 2
rest_client_requests_total{code="409",host="127.0.0.1:443",method="POST"} 3
rest_client_requests_total{code="409",host="127.0.0.1:443",method="PUT"} 8
# HELP ssh_tunnel_open_count Counter of ssh tunnel total open attempts
# TYPE ssh_tunnel_open_count counter
ssh_tunnel_open_count 0
# HELP ssh_tunnel_open_fail_count Counter of ssh tunnel failed open attempts
# TYPE ssh_tunnel_open_fail_count counter
ssh_tunnel_open_fail_count 0
```

Questo output non elaborato restituisce in modo completo ciò che il server API espone. I diversi parametri sono elencati per riga e ogni riga include un nome di parametro, tag e un valore.

```
metric_name{"tag"="value"[,...]}
            value
```

Supporto aggiuntivo Amazon EKS per Amazon CloudWatch

Amazon CloudWatch Observability raccoglie log, parametri e dati di traccia in tempo reale. Li invia ad Amazon CloudWatch e AWS X-Ray. Puoi installare questo componente aggiuntivo per abilitare sia CloudWatch Application Signals che CloudWatch Container Insights con una migliore osservabilità per Amazon EKS. In questo modo puoi monitorare lo stato e le prestazioni dell'infrastruttura e delle applicazioni containerizzate. Amazon CloudWatch Observability Operator è progettato per installare e configurare i componenti necessari.

Amazon EKS supporta Amazon CloudWatch Observability Operator come componente aggiuntivo di Amazon EKS. Il componente aggiuntivo lo consente sia Container Insights sui Windows nodi Linux di lavoro del cluster. Per Container Insights attivarloWindows, la versione del componente aggiuntivo Amazon EKS deve essere `1.5.0` o superiore. Attualmente, CloudWatch Application Signals non è supportato su Amazon EKSWindows.

Gli argomenti seguenti descrivono come iniziare a utilizzare Amazon CloudWatch Observability Operator per il cluster Amazon EKS.

- Per istruzioni sull'installazione di questo componente aggiuntivo, consulta Installare l' CloudWatch agente utilizzando i componenti aggiuntivi CloudWatch Amazon EKS di Observability nella Amazon CloudWatch User Guide.
- Per ulteriori informazioni su CloudWatch Application Signals, consulta Application Signals.
- Per ulteriori informazioniContainer Insights, consulta Using Container Insights in the Amazon CloudWatch User Guide.

Logging del piano di controllo di Amazon EKS

La registrazione del piano di controllo di Amazon EKS fornisce log di audit e diagnostica direttamente dal piano di controllo di Amazon CloudWatch EKS ai log del tuo account. Questi log consentono di semplificare la protezione e l'esecuzione dei cluster. Puoi selezionare i tipi di log esatti di cui hai bisogno e i log vengono inviati come flussi di log a un gruppo per ogni cluster Amazon EKS in cui opera. CloudWatch Per ulteriori informazioni, consulta Amazon CloudWatch logging.

É possibile iniziare a utilizzare il piano di controllo Amazon EKS scegliendo quali tipi di log che si desidera abilitare per ogni cluster Amazon EKS nuovo o esistente. È possibile abilitare o disabilitare ogni tipo di log in base al cluster utilizzando la AWS Management Console AWS CLI (versione `1.16.139` o superiore) o tramite l'API Amazon EKS. Se abilitato, i log vengono inviati automaticamente dal cluster Amazon CloudWatch EKS ai log nello stesso account.

Quando si utilizza la registrazione del piano di controllo Amazon EKS, viene addebitato il prezzo Amazon EKS standard per ogni cluster eseguito. Ti vengono addebitati i costi standard di inserimento e archiviazione dei dati di CloudWatch Logs per tutti i log inviati a Logs dai tuoi cluster. CloudWatch Inoltre, vengono addebitati i costi delle risorse AWS, ad esempio istanze Amazon EC2 o volumi Amazon EBS, di cui si effettua il provisionig come parte del cluster.

Sono disponibili i seguenti tipi di log del piano di controllo (control plane) del cluster. Ogni tipo di log corrisponde a un componente del piano di controllo di Kubernetes. Per ulteriori informazioni su questi componenti, consulta Componenti di Kubernetes nella documentazione di Kubernetes.

Server API (**`api`**)

> Il server API del cluster è il componente del piano di controllo (control-plane) che espone l'API di Kubernetes. Se abiliti i log del server API quando avvii il cluster o poco dopo, questi log includono

i flag del server API utilizzati per l'avvio del server API. Per ulteriori informazioni, consulta `kube-apiserver` e la policy di audit nella documentazione di Kubernetes.

Audit (**`audit`**)

I log di audit di Kubernetes forniscono un record dei singoli utenti, amministratori o componenti di sistema che hanno interessato il cluster. Per ulteriori informazioni, consulta Controllo nella documentazione di Kubernetes.

Autenticatore (**`authenticator`**)

I log dell'autenticatore sono univoci per Amazon EKS. Questi log rappresentano il componente del piano di controllo utilizzato da Amazon EKS per l'autenticazione del controllo degli accessi basati sui ruoli (RBAC) di Kubernetes utilizzando credenziali IAM. Per ulteriori informazioni, consulta Gestione dei cluster.

Gestore controller (**`controllerManager`**)

Il gestore controller gestisce i loop di controllo principali inviati con Kubernetes. Per ulteriori informazioni, consulta kube-controller-manager nella documentazione Kubernetes.

Pianificatore (**`scheduler`**)

Il componente pianificatore gestisce quando e dove eseguire i Pods nel cluster. Per ulteriori informazioni, consulta kube-scheduler nella documentazione di Kubernetes.

Abilitazione e disabilitazione dei log del piano di controllo

Per impostazione predefinita, i log del piano di controllo del cluster non vengono inviati ai Logs. CloudWatch È necessario abilitare ogni tipo di registro singolarmente per inviare i log per il cluster. CloudWatch Le tariffe di inserimento dei log, archiviazione, archiviazione e scansione dei dati si applicano ai log del piano di controllo abilitati. Per ulteriori informazioni, consulta la pagina dei prezzi. CloudWatch

Per aggiornare la configurazione di registrazione del piano di controllo (control-plane), Amazon EKS richiede fino a cinque indirizzi IP disponibili in ciascuna sottorete. Quando abiliti un tipo di registro, i log vengono inviati con un livello di dettaglio dei log di 2.

AWS Management Console

Per abilitare o disabilitare i log del piano di controllo con la AWS Management Console

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegliere il nome del cluster per visualizzare le informazioni sul cluster.
3. Scegli la scheda Osservabilità.
4. Nella sezione Registrazione del piano di controllo, scegli Gestisci registrazione.
5. Per ogni singolo tipo di log, scegli se il tipo di log deve essere attivato o disattivato. Per impostazione predefinita, i tipi di log sono disattivati.
6. Scegliere Salva le modifiche per terminare.

AWS CLI

Per abilitare o disabilitare i log del piano di controllo con la AWS CLI

1. Controllare la versione della AWS CLI con il seguente comando.

   ```
   aws --version
   ```

   Se la versione della AWS CLI è precedente alla `1.16.139`, dovrai prima eseguire l'aggiornamento alla versione più recente. Per installare o aggiornare il AWS CLI, consultare Installazione di AWS Command Line Interface nella AWS Command Line Interface Guida per l'utente.
2. Aggiornare la configurazione di esportazione dei log del piano di controllo del cluster con il comando della AWS CLI seguente. Sostituisci *`my-cluster`* con il nome del cluster e specifica i valori desiderati per l'accesso all'endpoint.

   > **Note**
   >
   > Il comando seguente invia tutti i tipi di log disponibili a CloudWatch Logs.

   ```
   aws eks update-cluster-config \
       --region region-code \
       --name my-cluster \
   ```

```
    --logging '{"clusterLogging":[{"types":
["api","audit","authenticator","controllerManager","scheduler"],"enabled":true}]}'
```

Di seguito viene riportato un output di esempio:

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "InProgress",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

3. Monitorare lo stato di aggiornamento della configurazione del log con il comando seguente utilizzando il nome del cluster e l'ID di aggiornamento restituiti dal comando precedente. L'aggiornamento è completo quando lo stato viene visualizzato come `Successful`.

```
aws eks describe-update \
    --region region-code\
    --name my-cluster \
    --update-id 883405c8-65c6-4758-8cee-2a7c1340a6d9
```

Di seguito viene riportato un output di esempio:

```
{
    "update": {
        "id": "883405c8-65c6-4758-8cee-2a7c1340a6d9",
        "status": "Successful",
        "type": "LoggingUpdate",
        "params": [
            {
                "type": "ClusterLogging",
```

```
                "value": "{\"clusterLogging\":[{\"types\":[\"api\",\"audit\",
\"authenticator\",\"controllerManager\",\"scheduler\"],\"enabled\":true}]}"
            }
        ],
        "createdAt": 1553271814.684,
        "errors": []
    }
}
```

Visualizzazione dei log del piano di controllo del cluster

Dopo aver abilitato uno qualsiasi dei tipi di log del piano di controllo per il tuo cluster Amazon EKS, puoi visualizzarli sulla CloudWatch console.

Per ulteriori informazioni sulla visualizzazione, l'analisi e la gestione dei log in CloudWatch, consulta la Amazon CloudWatch Logs User Guide.

Per visualizzare i log del piano di controllo del cluster sulla console CloudWatch

1. Apri la CloudWatch console. Questo link apre la console e mostra i gruppi di log disponibili correnti e li filtra con il prefisso `/aws/eks`.
2. Scegliere il cluster per il quale si intende visualizzare i log. Il formato del nome del gruppo di log è `/aws/eks/my-cluster/cluster`.
3. Scegliere il flusso di log da visualizzare. L'elenco seguente descrive il formato del nome del flusso di log per ogni tipo di registro.

> **Note**
>
> Al crescere della quantità di dati del flusso di log, i nomi del flusso di log vengono ruotati. Quando esistono più flussi di log per un determinato tipo di log, puoi visualizzare l'ultimo flusso di log cercando il nome del flusso di log con Last event time (Ora ultimo evento) più recente.

- Log dei componenti del server API Kubernetes (**`api`**) - `kube-apiserver-1234567890abcdef01234567890abcde`
- Controllo (**`audit`**) – `kube-apiserver-audit-1234567890abcdef01234567890abcde`

- Autenticatore (**`authenticator`**) – `authenticator-`*`1234567890abcdef01234567890abcde`*
- Gestore controller (**`controllerManager`**) – `kube-controller-manager-`*`1234567890abcdef01234567890abcde`*
- Pianificatore (**`scheduler`**) – `kube-scheduler-`*`1234567890abcdef01234567890abcde`*

4. Esamina gli eventi del flusso di log.

   Ad esempio, dovresti vedere i flag iniziali del server API del cluster quando visualizzi la parte iniziale di `kube-apiserver-`*`1234567890abcdef01234567890abcde`*.

   > **Note**
   >
   > Se i log del server API non vengono visualizzati all'inizio del flusso di log, è probabile che il file di log del server API sia stato ruotato sul server prima di abilitare la registrazione del server API sul server. I file di log che vengono ruotati prima dell'attivazione della registrazione del server API non possono essere esportati in. CloudWatch Tuttavia, potrai creare un nuovo cluster con la stessa versione di Kubernetes e abilitare la registrazione del server API quando si crea il cluster. I cluster con la stessa versione della piattaforma hanno gli stessi flag abilitati, quindi i flag devono corrispondere ai flag del nuovo cluster. Al termine della visualizzazione dei flag relativi al nuovo cluster CloudWatch, è possibile eliminare il nuovo cluster.

Registrazione delle chiamate API di Amazon EKS con AWS CloudTrail

Amazon EKS è integrato con AWS CloudTrail. CloudTrail è un servizio che fornisce un registro delle operazioni eseguite da un utente, un ruolo o un servizio AWS in Amazon EKS. CloudTrail acquisisce tutte le chiamate API per Amazon EKS come eventi. Questi includono le chiamate della console Amazon RDS e le chiamate del codice alle operazioni API Amazon EKS.

Se crei un percorso, puoi abilitare la distribuzione continua di eventi CloudTrail in un bucket Amazon S3, inclusi gli eventi per Amazon EKS. Se non configuri un percorso, puoi comunque visualizzare gli eventi più recenti nella console di CloudTrail in Cronologia eventi. Utilizzando le informazioni raccolte da CloudTrail, è possibile determinare diversi dettagli su una richiesta. Ad esempio, puoi

determinare quando è stata effettuata una richiesta ad Amazon EKS, l'indirizzo IP da cui è stata eseguita e l'autore.

Per ulteriori informazioni su CloudTrail, consultare la AWS CloudTrailGuida per l'utente di .

Argomenti

- Informazioni su Amazon EKS in CloudTrail
- Informazioni sulle voci del file di log Amazon EKS
- Abilitazione della raccolta dei parametri di un gruppo con scalabilità automatica

Informazioni su Amazon EKS in CloudTrail

Al momento della creazione del tuo account AWS, CloudTrail è abilitato sul tuo account AWS. Quando si verifica un'attività in Amazon EKS, questa viene registrata in un evento CloudTrail insieme ad altri eventi di servizio AWS nella Cronologia eventi. È possibile visualizzare, cercare e scaricare gli eventi recenti nell'account AWS. Per ulteriori informazioni, consulta Visualizzazione di eventi mediante la cronologia eventi di CloudTrail.

Per una registrazione continua degli eventi nell'account AWS, inclusi gli eventi per Amazon EKS, creare un percorso. Un percorso consente a CloudTrail di distribuire i file di log in un bucket Amazon S3. Per impostazione predefinita, quando si crea un percorso nella console, questo sarà valido in tutte le Regioni AWS. Il percorso registra gli eventi da tutte le Regioni AWS nella partizione AWS e distribuisce i file di log nel bucket Amazon S3 specificato. Inoltre, è possibile configurare altri servizi AWS per analizzare con maggiore dettaglio e usare i dati evento raccolti nei registri CloudTrail. Per ulteriori informazioni, consulta le risorse seguenti.

- Panoramica della creazione di un percorso
- Servizi e integrazioni CloudTrail supportati
- Configurazione delle notifiche Amazon SNS per CloudTrail
- Ricezione di file di registro CloudTrail da più Regioni e Ricezione di file di log CloudTrail da più account

Tutte le operazioni Amazon EKS vengono registrate da CloudTrail e sono documentate nella Documentazione di riferimento delle API di Amazon EKS. Ad esempio, le chiamate alle sezioni `CreateCluster`, `ListClusters` e `DeleteCluster` generano voci nei file di log CloudTrail.

Ogni evento o voce di log include informazioni sul tipo di identità IAM che ha effettuato la richiesta e sulle credenziali utilizzate. Se sono state utilizzate credenziali temporanee, la voce mostra il modo in cui tali credenziali sono state ottenute.

Per ulteriori informazioni, consulta Elemento CloudTrail userIdentity.

Informazioni sulle voci del file di log Amazon EKS

Un percorso è una configurazione che consente la distribuzione di eventi come i file di log in un bucket Amazon S3 specificato. I file di log di CloudTrail possono contenere una o più voci di log. Un evento rappresenta una singola richiesta da un'origine e include informazioni sull'operazione richiesta, come data e ora dell'operazione e i parametri della richiesta. I file di log CloudTrail non sono una traccia di pila ordinata delle chiamate API pubbliche e di conseguenza non devono apparire in base a un ordine specifico.

L'esempio seguente mostra una voce di registro di CloudTrail che illustra l'operazione `CreateCluster`.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AKIAIOSFODNN7EXAMPLE",
    "arn": "arn:aws:iam::111122223333:user/username",
    "accountId": "111122223333",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName": "username"
  },
  "eventTime": "2018-05-28T19:16:43Z",
  "eventSource": "eks.amazonaws.com",
  "eventName": "CreateCluster",
  "awsRegion": "region-code",
  "sourceIPAddress": "205.251.233.178",
  "userAgent": "PostmanRuntime/6.4.0",
  "requestParameters": {
    "resourcesVpcConfig": {
      "subnetIds": [
        "subnet-a670c2df",
        "subnet-4f8c5004"
      ]
    },
```

```
    "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
    "clusterName": "test"
  },
  "responseElements": {
    "cluster": {
      "clusterName": "test",
      "status": "CREATING",
      "createdAt": 1527535003.208,
      "certificateAuthority": {},
      "arn": "arn:aws:eks:region-code:111122223333:cluster/test",
      "roleArn": "arn:aws:iam::111122223333:role/AWSServiceRoleForAmazonEKS-
CAC1G1VH3ZKZ",
      "version": "1.10",
      "resourcesVpcConfig": {
        "securityGroupIds": [],
        "vpcId": "vpc-21277358",
        "subnetIds": [
          "subnet-a670c2df",
          "subnet-4f8c5004"
        ]
      }
    }
  },
  "requestID": "a7a0735d-62ab-11e8-9f79-81ce5b2b7d37",
  "eventID": "eab22523-174a-499c-9dd6-91e7be3ff8e3",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```


Voci di registro per ruoli collegati ai servizi Amazon EKS

I ruoli collegati ai servizi Amazon EKS effettuano chiamate API alle risorse AWS. Verranno visualizzate le voci di registro CloudTrail con `username: AWSServiceRoleForAmazonEKS` e `username: AWSServiceRoleForAmazonEKSNodegroup` per chiamate effettuate dai ruoli collegati ai servizi Amazon EKS. Per ulteriori informazioni su Amazon EKS e sui ruoli collegati ai servizi, consultare Utilizzo di ruoli collegati ai servizi per Amazon EKS.

Nell'esempio seguente viene illustrata una voce di registro CloudTrail che illustra un' operazione `DeleteInstanceProfile` eseguita dal ruolo collegato ai servizi `AWSServiceRoleForAmazonEKSNodegroup`, annotato nel `sessionContext`.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3WHGPEZ7SJ2CW55C5:EKS",
        "arn": "arn:aws:sts::111122223333:assumed-role/
AWSServiceRoleForAmazonEKSNodegroup/EKS",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROA3WHGPEZ7SJ2CW55C5",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/eks-
nodegroup.amazonaws.com/AWSServiceRoleForAmazonEKSNodegroup",
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAmazonEKSNodegroup"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2020-02-26T00:56:33Z"
            }
        },
        "invokedBy": "eks-nodegroup.amazonaws.com"
    },
    "eventTime": "2020-02-26T00:56:34Z",
    "eventSource": "iam.amazonaws.com",
    "eventName": "DeleteInstanceProfile",
    "awsRegion": "region-code",
    "sourceIPAddress": "eks-nodegroup.amazonaws.com",
    "userAgent": "eks-nodegroup.amazonaws.com",
    "requestParameters": {
        "instanceProfileName": "eks-11111111-2222-3333-4444-abcdef123456"
    },
    "responseElements": null,
    "requestID": "11111111-2222-3333-4444-abcdef123456",
    "eventID": "11111111-2222-3333-4444-abcdef123456",
    "eventType": "AwsApiCall",
    "recipientAccountId": "111122223333"
}
```

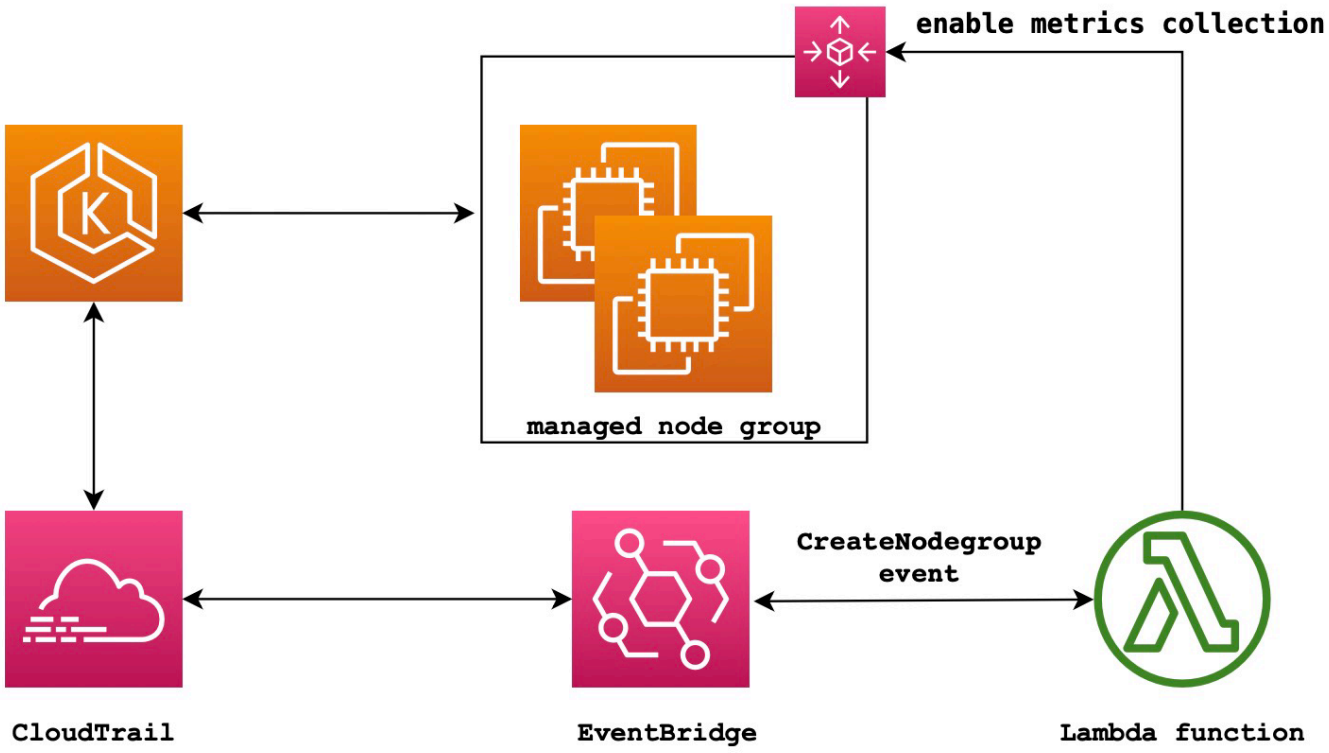
Abilitazione della raccolta dei parametri di un gruppo con scalabilità automatica

Questo argomento descrive come abilitare la raccolta dei parametri di un gruppo con scalabilità automatica utilizzando AWS Lambda e AWS CloudTrail. Amazon EKS non abilita automaticamente la raccolta dei parametri di gruppo per i gruppi con scalabilità automatica creati per nodi gestiti.

Puoi utilizzare i parametri di un gruppo con scalabilità automatica per tracciare le modifiche in un gruppo con scalabilità automatica e impostare allarmi sui valori soglia. Le metriche del gruppo Auto Scaling sono disponibili nella console Auto Scaling o nella console Amazon. CloudWatch Una volta abilitato, il gruppo Auto Scaling invia dati campionati ad Amazon CloudWatch ogni minuto. L'abilitazione di questi parametri non prevede alcun costo.

Abilitando la raccolta dei parametri di un gruppo con scalabilità automatica, potrai monitorare la scalabilità dei gruppi di nodi gestiti. I parametri di un gruppo con scalabilità automatica riportano la dimensione minima, massima e desiderata di un gruppo con scalabilità automatica. Puoi creare un allarme se il numero di nodi in un gruppo di nodi scende al di sotto della dimensione minima, fatto che indicherebbe un gruppo di nodi non integro. Il tracciamento delle dimensioni dei gruppi di nodi è utile anche per regolare il numero massimo in modo che il piano dati non esaurisca la capacità.

Quando crei un gruppo di nodi gestito, AWS CloudTrail invia un `CreateNodegroup` evento ad Amazon EventBridge. Creando una EventBridge regola Amazon che corrisponde all'`CreateNodegroup`evento, attivi una funzione Lambda per abilitare la raccolta di metriche di gruppo per il gruppo Auto Scaling associato al gruppo di nodi gestiti.



Come abilitare la raccolta dei parametri di un gruppo con scalabilità automatica

1. Crea un ruolo IAM per Lambda.

```
LAMBDA_ROLE=$(aws iam create-role \
  --role-name lambda-asg-enable-metrics \
  --assume-role-policy-document '{"Version": "2012-10-17","Statement":
 [{ "Effect": "Allow", "Principal": {"Service": "lambda.amazonaws.com"}, "Action":
 "sts:AssumeRole"}]}' \
  --output text \
  --query 'Role.Arn')
echo $LAMBDA_ROLE
```

2. Crea una policy che consenta di descrivere i gruppi di nodi Amazon EKS e abilitare la raccolta dei parametri di un gruppo con scalabilità automatica.

```
cat > /tmp/lambda-policy.json <<EOF
{
    "Version": "2012-10-17",
    "Statement": [
```

```
      {
        "Effect": "Allow",
        "Action": [
            "eks:DescribeNodegroup",
            "autoscaling:EnableMetricsCollection"
            ],
        "Resource": [
            "*"
        ]
      }
    ]
}
EOF
LAMBDA_POLICY_ARN=$(aws iam create-policy \
  --policy-name lambda-asg-enable-metrics-policy \
  --policy-document file:///tmp/lambda-policy.json \
  --output text \
  --query 'Policy.Arn')
echo $LAMBDA_POLICY_ARN
```

3. Collega la policy al ruolo IAM per Lambda.

```
aws iam attach-role-policy \
  --policy-arn $LAMBDA_POLICY_ARN \
  --role-name lambda-asg-enable-metrics
```

4. Aggiungi la politica `AWSLambdaBasicExecutionRole` gestita, che dispone delle autorizzazioni necessarie alla funzione per scrivere i log in Logs. CloudWatch

```
aws iam attach-role-policy \
  --role-name lambda-asg-enable-metrics \
  --policy-arn arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole
```

5. Crea il codice Lambda.

```
cat > /tmp/lambda-handler.py <<EOF
import json
import boto3
import time
import logging

eks = boto3.client('eks')
autoscaling = boto3.client('autoscaling')
```

```
logger = logging.getLogger()
logger.setLevel(logging.INFO)

def lambda_handler(event, context):
    ASG_METRICS_COLLLECTION_TAG_NAME = "ASG_METRICS_COLLLECTION_ENABLED"
    initial_retry_delay = 10
    attempts = 0

    #print(event)

    if not event["detail"]["eventName"] == "CreateNodegroup":
        print("invalid event.")
        return -1

    clusterName = event["detail"]["requestParameters"]["name"]
    nodegroupName = event["detail"]["requestParameters"]["nodegroupName"]
    try:
        metricsCollectionEnabled = event["detail"]["requestParameters"]["tags"]
[ASG_METRICS_COLLLECTION_TAG_NAME]
    except KeyError:
        print(ASG_METRICS_COLLLECTION_TAG_NAME, "tag not found.")
        return

    # Check if metrics collection is enabled in tags
    if metricsCollectionEnabled.lower() != "true":
        print("Metrics collection is not enabled in nodegroup tags.")
        return

    # Get the name of the associated autoscaling group
    print("Getting the autoscaling group name for nodegroup=", nodegroupName, ",
 cluster=", clusterName )
    for i in range(0,10):
        try:
            autoScalingGroup =
 eks.describe_nodegroup(clusterName=clusterName,nodegroupName=nodegroupName)
["nodegroup"]["resources"]["autoScalingGroups"][0]["name"]
        except:
            attempts += 1
            print("Failed to obtain the associated autoscaling group for
 nodegroup", nodegroupName, "Retrying in", initial_retry_delay*attempts,
 "seconds.")
            time.sleep(initial_retry_delay*attempts)
        else:
```

```
            break

    print("Enabling metrics collection on autoscaling group ", autoScalingGroup)

    # Enable metrics collection in the autoscaling group
    try:
        enableMetricsCollection =
 autoscaling.enable_metrics_collection(AutoScalingGroupName=autoScalingGroup,Granularity="1
    except:
        print("Unable to enable metrics collection on nodegroup=",nodegroup)
    print("Enabled metrics collection on nodegroup", nodegroupName)
EOF
```

6. Crea un pacchetto di implementazione.

```
cd /tmp
zip function.zip lambda-handler.py
```

7. Creazione di una funzione Lambda.

```
LAMBDA_ARN=$(aws lambda create-function --function-name asg-enable-metrics-
collection \
  --zip-file fileb://function.zip --handler lambda-handler.lambda_handler \
  --runtime python3.9 \
  --timeout 600 \
  --role $LAMBDA_ROLE \
  --output text \
  --query 'FunctionArn')
echo $LAMBDA_ARN
```

8. Crea una regola. EventBridge

```
RULE_ARN=$(aws events put-rule --name CreateNodegroupRuleToLambda \
  --event-pattern "{\"source\":[\"aws.eks\"],\"detail-type\":[\"AWS API Call via
 CloudTrail\"],\"detail\":{\"eventName\":[\"CreateNodegroup\"],\"eventSource\":
[\"eks.amazonaws.com\"]}}" \
  --output text \
  --query 'RuleArn')
echo $RULE_ARN
```

9. Aggiungi la funzione Lambda come destinazione.

```
aws events put-targets --rule CreateNodegroupRuleToLambda \
```

```
  --targets "Id"="1","Arn"="$LAMBDA_ARN"
```

10. Aggiungi una politica che EventBridge consenta di richiamare la funzione Lambda.

```
aws lambda add-permission \
  --function-name asg-enable-metrics-collection \
  --statement-id CreateNodegroupRuleToLambda \
  --action 'lambda:InvokeFunction' \
  --principal events.amazonaws.com \
  --source-arn $RULE_ARN
```

La funzione Lambda abilita la raccolta dei parametri di un gruppo con scalabilità automatica per ogni gruppo di nodi gestiti con tag `ASG_METRICS_COLLLECTION_ENABLED` impostati su `TRUE`. Per accertarti che Auto Scaling group metrics collection (Raccolta dei parametri di un gruppo con scalabilità automatica) sia abilitata, accedi al gruppo con scalabilità automatica associato nella console Amazon EC2. Nella scheda Monitoring (Monitoraggio) dovresti vedere che la casella di controllo Enable (Attiva) è attivata.

Supporto aggiuntivo di Amazon EKS per l'operatore ADOT

Amazon EKS supporta ora l'utilizzo della AWS Management Console, della AWS CLI e dell'API di Amazon EKS per installare e gestire l'operatore AWS Distro per OpenTelemetry (ADOT). Ciò rende più semplice la possibilità per le applicazioni in esecuzione su Amazon EKS di inviare parametri e dati di traccia a vari servizi di monitoraggio, ad esempio Amazon CloudWatch, Prometheus e X-Ray.

Per ulteriori informazioni, consulta Nozioni di base su AWS Distro per OpenTelemetry con i componenti aggiuntivi di EKS nella documentazione di AWS Distro per OpenTelemetry.

Altri AWS servizi integrati con Amazon EKS

Oltre ai servizi descritti in altre sezioni, Amazon EKS collabora con altri AWS servizi per fornire soluzioni aggiuntive. Questo argomento identifica alcuni degli altri servizi che utilizzano Amazon EKS per aggiungere funzionalità o i servizi utilizzati da Amazon EKS per eseguire le attività.

Argomenti

Creazione di risorse Amazon EKS con AWS CloudFormation

Amazon EKS è integrato con AWS CloudFormation, un servizio che consente di modellare e configurare le risorse AWS in modo da dedicare meno tempo alla creazione e alla gestione delle risorse e dell'infrastruttura. É possibile creare un modello che descriva tutte le risorse AWS che desideri, ad esempio un cluster Amazon EKS. In questo modo AWS CloudFormation gestirà il provisioning e la configurazione delle risorse per te.

Quando si utilizza AWS CloudFormation, è possibile riutilizzare il modello per configurare le risorse Amazon EKS in modo coerente e continuo. È sufficiente descrivere le risorse una volta e quindi allestire le stesse risorse più volte in più regioni e account AWS.

Amazon EKS e modelli AWS CloudFormation

É necessario conoscere i modelli AWS CloudFormation, prima di poter effettuare il provisioning e la configurazione delle risorse per Amazon EKS e per i servizi correlati. I modelli sono file di testo formattati in JSON o YAML. Questi modelli descrivono le risorse di cui intendi effettuare il provisioning negli stack AWS CloudFormation. Se non hai familiarità con JSON o YAML, puoi usare AWS CloudFormation Designer per iniziare a utilizzare i modelli AWS CloudFormation. Per ulteriori

informazioni, consulta Che cos'è AWS CloudFormation Designer? nella Guida per l'utente di AWS CloudFormation.

Amazon EKS supporta la creazione di cluster e gruppi di nodi in AWS CloudFormation. Per ulteriori informazioni, inclusi esempi di modelli JSON e YAML per le risorse Amazon EKS, consulta Riferimento ai tipi di risorse Amazon EKS nella AWS CloudFormation Guida per l'utente.

Ulteriori informazioni su AWS CloudFormation

Per ulteriori informazioni su AWS CloudFormation, consulta le seguenti risorse:

- AWS CloudFormation
- Guida per l'utente di AWS CloudFormation
- Guida per l'utente dell'interfaccia a riga di comando di AWS CloudFormation

Amazon EKS e zone locali AWS

Una zona locale AWS è un'estensione di una Regione AWS in prossimità geografica degli utenti. Le zone locali hanno le loro connessioni a Internet e supportano AWS Direct Connect. Le risorse create in una zona locale possono servire gli utenti locali con comunicazioni a latenza molto bassa. Per ulteriori informazioni, consulta Zone locali.

Amazon EKS supporta alcune risorse nelle zone locali, tra cui i nodi Amazon EC2 autogestiti, i volumi Amazon EBS e gli Application Load Balancer (ALB). Si consiglia di prendere in considerazione quanto segue nel momento in cui si utilizzano le zone locali come parte del cluster Amazon EKS.

Nodi

Non è possibile creare gruppi di nodi gestiti o nodi Fargate nelle zone locali con Amazon EKS. Tuttavia, è possibile creare nodi Amazon EC2 autogestiti nelle zone locali utilizzando l'API Amazon EC2, AWS CloudFormation o `eksctl`. Per ulteriori informazioni, consulta Nodi autogestiti.

Architettura di rete

- Il piano di controllo Kubernetes gestito da Amazon EKS viene sempre eseguito nella Regione AWS. Il piano di controllo Kubernetes gestito da Amazon EKS non può essere eseguito nella zona locale. Poiché le zone locali vengono visualizzate come sottoreti all'interno del VPC, Kubernetes considera le risorse della zona locale come parte di tale sottorete.

- Il cluster Kubernetes di Amazon EKS comunica con le istanze Amazon EC2 eseguite nella Regione AWS o nella zona locale utilizzando le interfacce di rete elastiche gestite da Amazon EKS. Per ulteriori informazioni sull'architettura di rete Amazon EKS, consulta Reti Amazon EKS.
- A differenza delle sottoreti regionali, Amazon EKS non può inserire interfacce di rete nelle sottoreti delle zone locali. Ciò significa che non è necessario specificare sottoreti della zona locale quando si crea il cluster.

Deep Learning Containers

I container AWS Deep Learning sono un set di immagini Docker per modelli di addestramento e fornitura in TensorFlow su Amazon EKS e Amazon Elastic Container Service (Amazon ECS). I Deep Learning Container forniscono ambienti ottimizzati con librerie TensorFlow, NVIDIA CUDA CUDA (per istanze GPU) e Intel MKL (per le istanze CPU) e sono disponibili in Amazon ECR.

Per iniziare a usare i Deep Learning Containers AWS su Amazon EKS, consulta Installazione di Amazon EKS nella AWS Guida per gli sviluppatori di Deep Learning Containers.

Amazon VPC Lattice

Amazon VPC Lattice è un servizio di rete di applicazioni completamente gestito integrato direttamente nell'infrastruttura di rete AWS che puoi utilizzare per connettere, proteggere e monitorare i tuoi servizi su più account e cloud privati virtuali (VPC). Con Amazon EKS, puoi sfruttare Amazon VPC Lattice tramite l'uso del Controller API Gateway AWS, un'implementazione dell'API Gateway Kubernetes. Utilizzando Amazon VPC Lattice, puoi configurare la connettività tra cluster con semantica Kubernetes standard in modo semplice e coerente. Per iniziare a usare Amazon VPC Lattice con Amazon EKS, consulta la Guida per l'utente del controller API Gateway AWS.

AWS Resilience Hub

AWS Resilience Hub valuta la resilienza di un cluster Amazon EKS analizzandone l'infrastruttura. AWS Resilience Hub utilizza la configurazione relativa al controllo degli accessi basato sul ruolo (RBAC) di Kubernetes per valutare i carichi di lavoro Kubernetes implementati nel cluster. Per ulteriori informazioni, consulta la sezione Abilitazione dell'accesso AWS Resilience Hub per gli account di servizio nel cluster nella Guida per l'utente di AWS Resilience Hub.

Rileva le minacce con Amazon GuardDuty

Amazon GuardDuty è un servizio di rilevamento delle minacce che aiuta a proteggere account, contenitori, carichi di lavoro e dati nel tuo AWS ambiente. Utilizzando modelli di machine learning (ML) e funzionalità di rilevamento di anomalie e minacce, monitora GuardDuty continuamente diverse fonti di log e attività di runtime per identificare e dare priorità ai potenziali rischi per la sicurezza e alle attività dannose nel tuo ambiente.

Tra le altre funzionalità, GuardDuty offre le seguenti due funzionalità che rilevano potenziali minacce ai cluster EKS: EKS Protection e Runtime Monitoring.

Protezione EKS

Questa funzionalità fornisce una copertura per il rilevamento delle minacce per aiutarti a proteggere i cluster Amazon EKS monitorando i log di Kubernetes controllo associati. Kubernetesi log di controllo registrano le azioni sequenziali all'interno del cluster, incluse le attività degli utenti, delle applicazioni che utilizzano l'KubernetesAPI e il piano di controllo. Ad esempio, GuardDuty può identificare che le API chiamate per alterare potenzialmente le risorse in un Kubernetes cluster sono state richiamate da un utente non autenticato.

Quando attivi EKS Protection, GuardDuty potrai accedere ai log di controllo di Amazon EKS solo per il rilevamento continuo delle minacce. Se GuardDuty identifica una potenziale minaccia per il cluster, genera un risultato del log di Kubernetes controllo associato di un tipo specifico. Per ulteriori informazioni sui tipi di risultati disponibili nei log di Kubernetes controllo, consulta i tipi di ricerca Kubernetes dei log di controllo nella Amazon GuardDuty User Guide.

Per ulteriori informazioni, consulta EKS Protection nella Amazon GuardDuty User Guide.

Monitoraggio del runtime

Questa funzionalità monitora e analizza gli eventi a livello di sistema operativo, di rete e di file per aiutarti a rilevare potenziali minacce in carichi di AWS lavoro specifici del tuo ambiente.

Quando abiliti il monitoraggio del runtime e installi l' GuardDuty agente nei tuoi cluster Amazon EKS, GuardDuty inizia a monitorare gli eventi di runtime associati a questo cluster. Se GuardDuty identifica una potenziale minaccia per il cluster, genera un risultato associato al Runtime Monitoring. Ad esempio, una minaccia può iniziare potenzialmente compromettendo un singolo contenitore che esegue un'applicazione web vulnerabile. Questa applicazione Web potrebbe disporre delle autorizzazioni di accesso ai contenitori e ai carichi di lavoro sottostanti. In questo

scenario, credenziali configurate in modo errato potrebbero potenzialmente portare a un accesso più ampio all'account e ai dati in esso archiviati.

Per configurare Runtime Monitoring, installi l' GuardDuty agente nel tuo cluster come componente aggiuntivo Amazon EKS. Per ulteriori informazioni sul componente aggiuntivo, consulta. Componenti aggiuntivi Amazon EKS disponibili da Amazon EKS

Per ulteriori informazioni, consulta Runtime Monitoring nella Amazon GuardDuty User Guide.

Utilizzo di Amazon Security Lake con Amazon EKS

Amazon Security Lake è un servizio di data lake di sicurezza completamente gestito che consente di centralizzare i dati di sicurezza da varie fonti, tra cui Amazon EKS. Integrando Amazon EKS con Security Lake, puoi ottenere informazioni più approfondite sulle attività eseguite sulle tue Kubernetes risorse e migliorare il livello di sicurezza dei tuoi cluster Amazon EKS.

> **Note**
>
> Per ulteriori informazioni sull'utilizzo di Security Lake con Amazon EKS e sulla configurazione delle origini dati, consulta la documentazione di Amazon Security Lake.

Vantaggi dell'utilizzo di Security Lake con Amazon Amazon EKS

Dati di sicurezza centralizzati: Security Lake raccoglie e centralizza automaticamente i dati di sicurezza dai cluster Amazon EKS, insieme ai dati di altri servizi, provider SaaS AWS , fonti locali e fonti di terze parti. Ciò fornisce una visione completa del tuo livello di sicurezza nell'intera organizzazione.

Formato dati standardizzato: Security Lake converte i dati raccolti nel formato Open Cybersecurity Schema Framework (OCSF), che è uno schema open source standard. Questa normalizzazione consente un'analisi e un'integrazione più semplici con altri strumenti e servizi di sicurezza.

Rilevamento delle minacce migliorato: analizzando i dati di sicurezza centralizzati, inclusi i log del piano di controllo di Amazon EKS, puoi rilevare attività potenzialmente sospette all'interno dei cluster Amazon EKS in modo più efficace. Questo aiuta a identificare e rispondere tempestivamente agli incidenti di sicurezza.

Gestione semplificata dei dati: Security Lake gestisce il ciclo di vita dei dati di sicurezza con impostazioni di conservazione e replica personalizzabili. Ciò semplifica le attività di gestione dei dati e garantisce la conservazione dei dati necessari per scopi di conformità e controllo.

Attivazione di Security Lake per Amazon EKS

Per iniziare a utilizzare Security Lake con Amazon EKS, segui questi passaggi:

1. Abilita la registrazione del piano di controllo Amazon EKS per i tuoi cluster EKS. Per istruzioni dettagliate, consulta Abilitazione e disabilitazione dei log del piano di controllo.
2. Aggiungi Amazon EKS Audit Logs come origine in Security Lake. Security Lake inizierà quindi a raccogliere informazioni approfondite sulle attività eseguite sulle risorse Kubernetes in esecuzione nei cluster EKS.
3. Configura le impostazioni di conservazione e replica per i tuoi dati di sicurezza in Security Lake in base alle tue esigenze.
4. Utilizza i dati OCSF normalizzati archiviati in Security Lake per la risposta agli incidenti, l'analisi della sicurezza e l'integrazione con altri AWS servizi o strumenti di terze parti. Ad esempio, puoi generare informazioni sulla sicurezza dai dati di Amazon Security Lake utilizzando Amazon OpenSearch Ingestion.

Analisi dei log EKS in Security Lake

Security Lake normalizza gli eventi di registro EKS nel formato OCSF, semplificando l'analisi e la correlazione dei dati con altri eventi di sicurezza. Puoi utilizzare vari strumenti e servizi, come Amazon Athena, Amazon o strumenti di analisi della sicurezza di terze parti, per interrogare e visualizzare i dati normalizzati. QuickSight

Per ulteriori informazioni sulla mappatura OCSF per gli eventi di registro EKS, consulta il riferimento alla mappatura nel repository OCSF. GitHub

Amazon Detective

Amazon Detective consente di analizzare, esaminare e identificare rapidamente la causa principale degli esiti di sicurezza o delle attività sospette. Detective raccoglie automaticamente i dati di registro dalle tue AWS risorse. Utilizza quindi il machine learning, l'analisi statistica e la teoria dei grafi per generare visualizzazioni che consentono di condurre indagini sulla sicurezza più rapide ed efficaci. Le aggregazioni di dati, i riepiloghi e il contesto predefiniti di Detective facilitano e velocizzano l'analisi

e la determinazione della natura e dell'estensione dei possibili problemi di sicurezza. Per ulteriori informazioni, consulta la Guida per l'utente di Amazon Detective.

Detective organizza Kubernetes AWS e analizza risultati come:

- Dettagli del cluster Amazon EKS, tra cui l'identità IAM che ha creato il cluster e il ruolo di servizio del cluster. Puoi esaminare l'attività AWS e l'attività delle Kubernetes API di queste identità IAM con Detective.
- Dettagli del container, come l'immagine e il contesto di sicurezza. Puoi esaminare i dettagli relativi ai Pods terminati.
- Attività API di Kubernetes, incluse le tendenze generali di attività API e i dettagli su chiamate API specifiche. Ad esempio, puoi mostrare il numero di chiamate API di Kubernetes riuscite e non riuscite emesse in un intervallo di tempo selezionato. Inoltre, la sezione sulle chiamate API appena osservate potrebbe essere utile per identificare attività sospette.

I log di audit di Amazon EKS sono un pacchetto di origini dati facoltativo che può essere aggiunto al grafico del comportamento di Detective. Puoi visualizzare i pacchetti di origine facoltativi disponibili e il rispettivo stato dal tuo account. Per ulteriori informazioni, consulta Log di audit di Amazon EKS per Detective nella Guida per l'utente di Amazon Detective.

Uso di Amazon Detective con Amazon EKS

Esame dei risultati relativi a un cluster Amazon EKS

Prima di poter esaminare i risultati, Detective deve essere abilitato per almeno 48 ore nello stesso luogo in Regione AWS cui si trova il cluster. Per ulteriori informazioni, consulta le sezioni Configurazione di Amazon Detective nella Guida per l'utente di Amazon Detective.

1. Apri la console Detective all'indirizzo https://console.aws.amazon.com/detective/.
2. Seleziona Cerca nel riquadro di navigazione a sinistra.
3. Seleziona Scegli il tipo, quindi seleziona il Cluster EKS.
4. Inserisci il nome o l'ARN del cluster e quindi scegli Cerca.
5. Nei risultati della ricerca, seleziona il nome del cluster del quale desideri visualizzare l'attività. Per ulteriori informazioni su ciò che puoi visualizzare, consulta Attività complessiva delle API di Kubernetes che interessano un cluster Amazon EKS nella Guida per l'utente di Amazon Detective.

Risoluzione dei problemi di Amazon EKS

Questo capitolo presenta alcuni errori comuni che si potrebbero verificare durante l'uso di Amazon EKS e il modo in cui gestirli. Se devi risolvere problemi in aree specifiche di Amazon EKS, consulta gli argomenti separati Risoluzione dei problemi di IAM, Risoluzione dei problemi in Amazon EKS Connector e Risoluzione dei problemi di ADOT utilizzando i componenti aggiuntivi EKS.

Per altre informazioni sulla risoluzione dei problemi, consulta i Contenuti del Knowledge Center su Amazon Elastic Kubernetes Service su AWS re:Post.

Capacità insufficiente

Se ricevi il seguente errore mentre provi a creare un cluster Amazon EKS, significa che una delle zone di disponibilità indicate non dispone della capacità sufficiente per supportare un cluster.

```
Cannot create cluster 'example-cluster' because region-1d, the targeted Availability Zone, does not currently have sufficient capacity to support the cluster. Retry and choose from these Availability Zones: region-1a, region-1b, region-1c
```

Riprova a crearlo con le sottoreti del VPC del cluster ospitate nelle zone di disponibilità restituite da questo messaggio di errore.

Ci sono zone di disponibilità dove non è possibile allocare un cluster. Confronta le zone di disponibilità in cui si trovano le tue sottoreti con l'elenco delle zone di disponibilità nella sezione Considerazioni e requisiti relativi alle sottoreti.

Impossibile aggiungere i nodi al cluster

Ci sono diversi motivi comuni che impediscono l'aggiunta dei nodi di lavoro al cluster:

- Se i nodi sono gestiti, Amazon EKS aggiunge voci ad `aws-auth ConfigMap` quando crei il gruppo di nodi. Se la voce è stata rimossa o modificata, è necessario aggiungerla nuovamente. Per ulteriori informazioni, digita **`eksctl create iamidentitymapping --help`** nel terminale. Puoi verificare la versione corrente delle voci `aws-auth ConfigMap` sostituendo *`my-cluster`* nel seguente comando con il nome del tuo cluster e quindi eseguendo il comando modificato: **`eksctl get iamidentitymapping --cluster`** ***`my-cluster`***. L'ARN del ruolo specificato non può

includere un percorso diverso da `/`. Ad esempio, se il nome del ruolo è `development/apps/my-role`, è necessario modificarlo in `my-role` quando si specifica l'ARN per tale ruolo. Assicurati di specificare l'ARN del ruolo IAM del nodo (non l'ARN del profilo dell'istanza).

Se i nodi sono autogestiti e non hai creato voci di accesso per l'ARN del ruolo IAM del nodo, esegui gli stessi comandi elencati per i nodi gestiti. Se hai creato una voce di accesso per l'ARN del tuo ruolo IAM del nodo, è possibile che non sia configurata correttamente nella suddetta voce di accesso. Assicurati che l'ARN del ruolo IAM del nodo (e non l'ARN del profilo dell'istanza) sia specificato come ARN principale nella tua voce `aws-auth ConfigMap` o voce di accesso. Per ulteriori informazioni sulle voci di accesso, consulta la sezione Gestire le voci di accesso.

- Il AWS CloudFormation modello ClusterNamenel tuo nodo non corrisponde esattamente al nome del cluster a cui desideri che i tuoi nodi si uniscano. Se inserisci un valore non corretto in questo campo, si verificherà una configurazione errata del file `/var/lib/kubelet/kubeconfig` del nodo e i nodi non verranno aggiunti al cluster.
- Il nodo non è taggato come appartenente al cluster. Ai nodi deve essere applicato il tag seguente, dove *`my-cluster`* viene sostituito con il nome del cluster.

Chiave	Valore
`kubernetes.io/cluster/` *`my-cluster`*	`owned`

- I nodi potrebbero non essere in grado di accedere al cluster utilizzando un indirizzo IP pubblico. Assicurarsi che ai nodi implementati nelle sottoreti pubbliche venga assegnato un indirizzo IP pubblico. In caso contrario, è possibile associare un indirizzo IP elastico a un nodo dopo l'avvio. Per ulteriori informazioni, consulta Associazione di un indirizzo IP elastico a un'istanza o a un'interfaccia di rete in esecuzione. Se nella sottorete pubblica non è impostata l'assegnazione degli indirizzi IP pubblici alle istanze implementate, è consigliabile abilitare l'impostazione. Per ulteriori informazioni, consulta Modifica dell'attributo di assegnazione degli indirizzi `IPv4` pubblici per la sottorete. Se il nodo viene implementato in una sottorete privata, la sottorete deve disporre di un instradamento verso un gateway NAT a cui è assegnato un indirizzo IP pubblico.
- L' AWS STS endpoint su Regione AWS cui stai distribuendo i nodi non è abilitato per il tuo account. Per abilitare la regione, vedi Attivazione e AWS STS disattivazione in un. Regione AWS
- Il nodo non dispone di una voce DNS privata, con conseguente log `kubelet` che riporta un errore `node "" not found`. Assicurati che il VPC in cui viene creato il nodo disponga di valori impostati per `domain-name` e `domain-name-servers` come `Options` in un `DHCP options`

`set`. I valori predefiniti sono `domain-name:<region>.compute.internal` e `domain-name-servers:AmazonProvidedDNS`. Per ulteriori informazioni, consulta Set opzioni DHCP nella Guida per l'utente di Amazon VPC.

- Se i nodi del gruppo di nodi gestiti non si connettono al cluster entro 15 minuti, verrà emesso un problema di integrità di «NodeCreationErrore» e lo stato della console verrà impostato su. `Create failed` Per le Windows AMI con tempi di avvio lenti, questo problema può essere risolto utilizzando l'avvio rapido.

Per identificare e risolvere i problemi più comuni che impediscono ai nodi worker di unirsi a un cluster, puoi utilizzare il runbook `AWSSupport-TroubleshootEKSWorkerNode`. Per ulteriori informazioni, consulta `AWSSupport-TroubleshootEKSWorkerNode` in Documentazione di riferimento del runbook di AWS Systems Manager Automation.

Accesso negato o non autorizzato (`kubectl`)

Se ricevi uno dei seguenti errori durante l'esecuzione di comandi `kubectl`, allora `kubectl` non è configurato correttamente per Amazon EKS, oppure le credenziali del principale IAM (ruolo o utente) in uso non sono mappate a un nome utente Kubernetes con autorizzazioni sufficienti per gli oggetti Kubernetes nel tuo cluster Amazon EKS.

- `could not get token: AccessDenied: Access denied`
- `error: You must be logged in to the server (Unauthorized)`
- `error: the server doesn't have a resource type "svc"`

Questo potrebbe essere dovuto a uno dei seguenti fattori:

- Il cluster è stato creato con le credenziali per un principale IAM e `kubectl` è configurato per utilizzare le credenziali per un altro principale IAM. Per risolvere il problema, aggiorna il file `kube config` affinché vengano utilizzate le credenziali con cui è stato creato il cluster. Per ulteriori informazioni, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.
- Se il cluster soddisfa i requisiti minimi di piattaforma indicati nella sezione dei prerequisiti di Gestire le voci di accesso, non esiste una voce di accesso con il tuo principale IAM. Se esiste, non ha i nomi dei gruppi Kubernetes necessari definiti per essa, o non ha associata la policy di accesso corretta. Per ulteriori informazioni, consulta Gestire le voci di accesso.

- Se il tuo cluster non soddisfa i requisiti minimi di piattaforma indicati in Gestire le voci di accesso, non esiste una voce con il tuo principale IAM in `aws-auth ConfigMap`. Se esiste, non è mappata su nomi dei gruppi Kubernetes associati a Kubernetes `Role` o `ClusterRole` con le autorizzazioni necessarie. Per ulteriori informazioni sugli oggetti del controllo degli accessi basato sui ruoli (RBAC) in Kubernetes, consulta Using RBAC authorization nella documentazione di Kubernetes. Puoi verificare la versione corrente delle voci `aws-auth ConfigMap` sostituendo *`my-cluster`* nel seguente comando con il nome del tuo cluster e quindi eseguendo il comando modificato: **`eksctl get iamidentitymapping --cluster`** ***`my-cluster`***. Se in `ConfigMap` non è presente una voce con l'ARN del tuo principale IAM, digita **`eksctl create iamidentitymapping --help`** nel terminale per scoprire come crearne una.

Se installi e configuri AWS CLI, puoi configurare le credenziali IAM che utilizzi. Per ulteriori informazioni, consulta Configurazione della AWS CLI nella Guida per l'utente di AWS Command Line Interface . Puoi anche configurare `kubectl` affinché utilizzi un ruolo IAM, se assumi un ruolo IAM per accedere agli oggetti Kubernetes sul tuo cluster. Per ulteriori informazioni, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.

`hostname doesn't match`

La versione Python del sistema deve essere `2.7.9` o successiva. In caso contrario, riceverai `hostname doesn't match` errori nelle AWS CLI chiamate ad Amazon EKS. Per ulteriori informazioni, consulta What are "hostname doesn't match" errors? nelle domande frequenti di Python Requests.

`getsockopt: no route to host`

Docker viene eseguito nell'intervallo CIDR `172.17.0.0/16` nei cluster Amazon EKS. Consigliamo che le sottoreti VPC del cluster non si sovrappongano a questo intervallo. In caso contrario, riceverai il seguente messaggio di errore:

```
Error: : error upgrading connection: error dialing backend: dial tcp
 172.17.<nn>.<nn>:10250: getsockopt: no route to host
```

`Instances failed to join the Kubernetes cluster`

Se ricevi l'errore `Instances failed to join the Kubernetes cluster` in AWS Management Console, assicurati che l'accesso privato agli endpoint del cluster sia abilitato o di aver configurato correttamente i blocchi CIDR per l'accesso pubblico agli endpoint. Per ulteriori informazioni, consulta Controllo accessi all'endpoint del cluster Amazon EKS.

Codici di errore dei gruppi di nodi gestiti

Se il gruppo di nodi gestiti rileva un problema di integrità hardware, Amazon EKS restituisce un codice di errore che consente di diagnosticare il problema. Questi controlli dell'integrità non rilevano problemi software perché si basano su controlli dell'integrità di Amazon EC2. Nel seguente elenco vengono descritti i codici di errore.

AccessDenied

Amazon EKS o uno o più nodi gestiti non sono in grado di eseguire l'autenticazione o l'autorizzazione con il server API del cluster Kubernetes. Per ulteriori informazioni sulla risoluzione di un problema comune, consulta Risoluzione di una causa comune di errori `AccessDenied` per i gruppi di nodi gestiti. Le AMI Windows private possono causare anche questo codice di errore insieme al messaggio di errore `Not authorized for images`. Per ulteriori informazioni, consulta `Not authorized for images`.

AmiIdNotFound

Non è stato possibile trovare l'ID AMI associato al modello di avvio. Assicurati che l'AMI esista e sia condivisa con l'account.

AutoScalingGroupNotTrovato

Non è stato possibile trovare il gruppo con dimensionamento automatico associato al gruppo di nodi gestiti. Potresti ricreare un gruppo con dimensionamento automatico con le stesse impostazioni per procedere con il ripristino.

ClusterUnreachable

Amazon EKS o uno o più nodi gestiti non sono in grado di comunicare con il server API del cluster Kubernetes. Ciò può verificarsi in caso di interruzioni di rete o se i server API stanno eseguendo il timeout delle richieste di elaborazione.

Ec2 SecurityGroup NotFound

Non è stato possibile trovare il gruppo di sicurezza per il cluster. È necessario ricreare il cluster.

Ec 2 SecurityGroup DeletionFailure

Non è stato possibile eliminare il gruppo di sicurezza dell'accesso remoto per il gruppo di nodi gestiti. Rimuovi eventuali dipendenze dal gruppo di sicurezza.

Ec 2 LaunchTemplate NotFound

Non è stato possibile trovare il modello di avvio Amazon EC2 per il gruppo di nodi gestiti. È necessario ricreare il gruppo di nodi per procedere con il ripristino.

Ec 2 LaunchTemplate VersionMismatch

La versione del modello di avvio Amazon EC2 per il gruppo di nodi gestiti non corrisponde alla versione creata da Amazon EKS. Puoi regredire alla versione creata da Amazon EKS per il ripristino.

IamInstanceProfileNotTrovato

Non è stato possibile trovare il profilo dell'istanza IAM per il gruppo di nodi gestiti. Potresti ricreare un profilo dell'istanza con le stesse impostazioni per il ripristino.

IamNodeRoleNotTrovato

Non è stato possibile trovare il ruolo IAM per il gruppo di nodi gestiti. Potresti ricreare un ruolo IAM con le stesse impostazioni per il ripristino.

AsgInstanceLaunchFailures

Il gruppo con dimensionamento automatico sta riscontrando errori nel tentativo di avviare le istanze.

NodeCreationFallimento

Le istanze avviate non sono in grado di registrarsi con il cluster Amazon EKS. Le cause comuni di questo errore sono le autorizzazioni insufficienti del ruolo IAM del nodo o la mancanza di accesso a Internet in uscita per i nodi. I nodi devono soddisfare uno dei requisiti seguenti:

- Devono essere in grado di accedere a Internet utilizzando un indirizzo IP pubblico. Il gruppo di sicurezza associato alla sottorete in cui si trova il nodo deve consentire la comunicazione. Per ulteriori informazioni, consulta Considerazioni e requisiti relativi alle sottoreti e Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS.

- I nodi e il VPC devono soddisfare i requisiti in Requisiti dei cluster privati.

InstanceLimitSuperato

Il tuo AWS account non è in grado di avviare altre istanze del tipo di istanza specificato. Potresti richiedere un aumento del limite di istanze Amazon EC2 per il ripristino.

InsufficientFreeIndirizzi

Una o più sottoreti associate al gruppo di nodi gestiti non dispone di indirizzi IP disponibili sufficienti per i nuovi nodi.

InternalFailure

Questi errori sono in genere causati da un problema lato server Amazon EKS.

Risoluzione di una causa comune di errori **`AccessDenied`** per i gruppi di nodi gestiti

La causa più comune degli errori `AccessDenied` durante l'esecuzione di operazioni su gruppi di nodi gestiti è la mancanza di `eks:node-manager` `ClusterRole` o `ClusterRoleBinding`. Amazon EKS imposta queste risorse nel cluster come parte dell'onboarding con i gruppi di nodi gestiti e queste sono necessarie per la gestione dei gruppi di nodi.

Il `ClusterRole` può cambiare nel tempo, ma dovrebbe essere simile all'esempio seguente:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
```

```
  - get
  - list
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
```

Il `ClusterRoleBinding` può cambiare nel tempo, ma dovrebbe essere simile all'esempio seguente:

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
- apiGroup: rbac.authorization.k8s.io
  kind: User
  name: eks:node-manager
```

Verifica che il `eks:node-manager` `ClusterRole` esista.

```
kubectl describe clusterrole eks:node-manager
```

Se presente, confrontare l'output con il precedente esempio `ClusterRole`.

Verifica che il `eks:node-manager` `ClusterRoleBinding` esista.

```
kubectl describe clusterrolebinding eks:node-manager
```

Se presente, confrontare l'output con il precedente esempio `ClusterRoleBinding`.

Se viene identificato un oggetto mancante o corrotto `ClusterRole` o `ClusterRoleBinding` come causa di un errore `AcessDenied` durante la richiesta di operazioni di gruppo di nodi gestiti,

è possibile ripristinarle. Salva nel computer i seguenti contenuti in un file denominato *eks-node-manager-role.yaml*.

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eks:node-manager
rules:
- apiGroups:
  - ''
  resources:
  - pods
  verbs:
  - get
  - list
  - watch
  - delete
- apiGroups:
  - ''
  resources:
  - nodes
  verbs:
  - get
  - list
  - watch
  - patch
- apiGroups:
  - ''
  resources:
  - pods/eviction
  verbs:
  - create
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: eks:node-manager
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eks:node-manager
subjects:
- apiGroup: rbac.authorization.k8s.io
```

```
  kind: User
  name: eks:node-manager
```

Applica il file.

```
kubectl apply -f eks-node-manager-role.yaml
```

Riprova l'operazione per il gruppo di nodi per verificare se il problema è stato risolto.

`Not authorized for images`

Una potenziale causa di un messaggio di errore `Not authorized for images` è l'uso di un'AMI Windows di Amazon EKS privata per avviare gruppi di nodi gestiti da Windows. Dopo aver rilasciato nuove Windows AMI, AWS rende private le AMI più vecchie di 4 mesi, il che le rende non più accessibili. Se il tuo gruppo di nodi gestiti utilizza un'WindowsAMI privata, valuta la possibilità di aggiornare il gruppo di nodi Windows gestiti. Sebbene non possiamo garantire di poter fornire l'accesso alle AMI che sono state rese private, puoi richiedere l'accesso inviando un ticket a Support AWS . Per ulteriori informazioni, consulta Patch, aggiornamenti di sicurezza e ID AMI nella Guida per l'utente di Amazon EC2.

Il nodo è in stato `NotReady`

Se il nodo entra in uno `NotReady` stato, è probabile che ciò indichi che il nodo non è integro e non è possibile pianificarne uno nuovoPods. Ciò può verificarsi per vari motivi, ad esempio se il nodo non dispone di risorse sufficienti per la CPU, la memoria o lo spazio disponibile su disco.

Per le Windows AMI ottimizzate per Amazon EKS, non è prevista alcuna prenotazione per le risorse di calcolo specificate di default nella `kubelet` configurazione. Per aiutare a prevenire problemi di risorse, puoi riservare risorse di calcolo per i processi di sistema fornendo loro valori `kubelet` di configurazione per e/o. `kube-reservedsystem-reserved` È possibile eseguire questa operazione utilizzando il parametro della `-KubeletExtraArgs` riga di comando nello script bootstrap. Per ulteriori informazioni, consulta Reserve Compute Resources for System Daemons nella Kubernetes documentazione e i parametri di configurazione dello script Bootstrap in questa guida per l'utente.

Strumento di raccolta di log CNI

Il Amazon VPC CNI plugin for Kubernetes ha il proprio script di risoluzione dei problemi che è disponibile sui nodi in `/opt/cni/bin/aws-cni-support.sh`. È possibile utilizzare lo script per raccogliere i registri diagnostici per i casi di supporto e per la risoluzione dei problemi generali.

Utilizza il comando seguente per eseguire lo script sul nodo:

```
sudo bash /opt/cni/bin/aws-cni-support.sh
```

> **Note**
>
> Se lo script non è presente in quella posizione, significa che l'esecuzione del container CNI non è andata a buon fine. Puoi scaricare manualmente lo script ed eseguirlo con il comando seguente:
>
> ```
> curl -O https://raw.githubusercontent.com/awslabs/amazon-eks-ami/master/log-collector-script/linux/eks-log-collector.sh
> sudo bash eks-log-collector.sh
> ```

Lo script raccoglie le seguenti informazioni di diagnostica: La versione CNI implementata può essere precedente alla versione dello script.

```
      This is version 0.6.1. New versions can be found at https://github.com/awslabs/amazon-eks-ami

Trying to collect common operating system logs...
Trying to collect kernel logs...
Trying to collect mount points and volume information...
Trying to collect SELinux status...
Trying to collect iptables information...
Trying to collect installed packages...
Trying to collect active system services...
Trying to collect Docker daemon information...
Trying to collect kubelet information...
Trying to collect L-IPAMD information...
Trying to collect sysctls information...
Trying to collect networking information...
Trying to collect CNI configuration information...
```

```
Trying to collect running Docker containers and gather container data...
Trying to collect Docker daemon logs...
Trying to archive gathered information...

 Done... your bundled logs are located in /var/
log/eks_i-0717c9d54b6cfaa19_2020-03-24_0103-UTC_0.6.1.tar.gz
```

Le informazioni di diagnostica vengono raccolte e archiviate in:

```
/var/log/eks_i-0717c9d54b6cfaa19_2020-03-24_0103-UTC_0.6.1.tar.gz
```

Rete runtime container non pronta

Potresti visualizzare un errore `Container runtime network not ready` ed errori di autorizzazione analoghi al seguente:

```
4191 kubelet.go:2130] Container runtime network not ready: NetworkReady=false
 reason:NetworkPluginNotReady message:docker: network plugin is not ready: cni config
 uninitialized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
4191 kubelet_node_status.go:106] Unable to register node
 "ip-10-40-175-122.ec2.internal" with API server: Unauthorized
4191 reflector.go:205] k8s.io/kubernetes/pkg/kubelet/kubelet.go:452: Failed to list
 *v1.Service: Unauthorized
```

Questo può verificarsi per uno dei seguenti motivi:

1. Non disponi di `aws-auth ConfigMap` sul tuo cluster oppure non include voci per il ruolo IAM con cui hai configurato i tuoi nodi.

   Questa voce `ConfigMap` è necessaria se i nodi soddisfano uno dei seguenti criteri:

   - Nodi gestiti in un cluster con qualsiasi versione di Kubernetes o della piattaforma.
   - Nodi autogestiti in un cluster con una versione precedente a una delle versioni della piattaforma elencate nella sezione dei prerequisiti dell'argomento Gestire le voci di accesso.

   Per risolvere il problema, verifica le voci esistenti in `ConfigMap` sostituendo *`my-cluster`* nel seguente comando con il nome del tuo cluster e quindi eseguendo il comando modificato: **`eksctl get iamidentitymapping --cluster my-cluster`**. Se ricevi un messaggio di errore dal comando, è possibile che il cluster non disponga di `aws-auth ConfigMap`. Il comando seguente

aggiunge una voce a `ConfigMap`. Inoltre, se `ConfigMap` non esiste, viene creata dal comando. Sostituisci *111122223333* con l' Account AWS ID per il ruolo IAM e *myAmazoneks NodeRole* con il nome del ruolo del tuo nodo.

```
eksctl create iamidentitymapping --cluster my-cluster \
    --arn arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --group
 system:bootstrappers,system:nodes \
    --username system:node:{{EC2PrivateDNSName}}
```

L'ARN del ruolo specificato non può includere un percorso diverso da `/`. Ad esempio, se il nome del tuo ruolo è `development/apps/my-role`, dovresti cambiarlo in `my-role` quando specifichi l'ARN del ruolo. Assicurati di specificare l'ARN del ruolo IAM del nodo (non l'ARN del profilo dell'istanza).

2. I nodi autogestiti si trovano in un cluster con una versione della piattaforma che corrisponde a quella minima elencata nei prerequisiti dell'argomento Gestire le voci di accesso, tuttavia una voce non è elencata in `aws-auth` `ConfigMap` (vedi l'elemento precedente) per il ruolo IAM del nodo oppure non esiste una voce di accesso per il ruolo. Per risolvere il problema, verifica le tue voci di accesso esistenti sostituendo *my-cluster* nel seguente comando con il nome del tuo cluster e quindi eseguendo il comando modificato: **aws eks list-access-entries --cluster-name *my-cluster***. Il comando seguente aggiunge una voce di accesso per il ruolo IAM del nodo. Sostituisci *111122223333* con l' Account AWS ID per il ruolo IAM e *NodeRolemyAmazoneks* con il nome del ruolo del tuo nodo. Se hai un nodo Windows, sostituisci *EC2_Linux* con **EC2_Windows**. Assicurati di specificare l'ARN del ruolo IAM del nodo (non l'ARN del profilo dell'istanza).

```
aws eks create-access-entry --cluster-name my-cluster --principal-arn
 arn:aws:iam::111122223333:role/myAmazonEKSNodeRole --type EC2_Linux
```

Timeout dell'handshake TLS

Quando un nodo non è in grado di stabilire una connessione all'endpoint del server API pubblico, è possibile che venga visualizzato un errore analogo al seguente.

```
server.go:233] failed to run Kubelet: could not init cloud provider "aws": error
 finding instance i-1111f2222f333e44c: "error listing AWS instances: \"RequestError:
 send request failed\\ncaused by: Post  net/http: TLS handshake timeout\""
```

Il processo `kubelet` riprenderà continuamente e testerà l'endpoint del server API. L'errore può verificarsi anche temporaneamente durante qualsiasi procedura che esegue un aggiornamento in sequenza del cluster nel piano di controllo, ad esempio una modifica della configurazione o un aggiornamento della versione.

Per risolvere il problema, controlla la tabella di routing e i gruppi di sicurezza per assicurarti che il traffico proveniente dai nodi possa raggiungere l'endpoint pubblico.

InvalidClientTokenId

Se utilizzi i ruoli IAM per gli account di servizio di un cluster in Cina Pod o li hai `DaemonSet` distribuiti in un cluster in Cina Regione AWS e non hai impostato la variabile di `AWS_DEFAULT_REGION` ambiente nelle specifiche, il sistema operativo potrebbe ricevere il seguente errore: Pod `DaemonSet`

```
An error occurred (InvalidClientTokenId) when calling the GetCallerIdentity operation:
 The security token included in the request is invalid
```

Per risolvere il problema, è necessario aggiungere la variabile di ambiente `AWS_DEFAULT_REGION` alla specifica del Pod o di `DaemonSet`, come mostrato nella specifica del Pod di esempio seguente.

```
apiVersion: v1
kind: Pod
metadata:
  name: envar-demo
  labels:
    purpose: demonstrate-envars
spec:
  containers:
  - name: envar-demo-container
    image: gcr.io/google-samples/node-hello:1.0
    env:
    - name: AWS_DEFAULT_REGION
      value: "region-code"
```

Scadenza del certificato webhook di ammissione del VPC

Se il certificato utilizzato per firmare il webhook di ammissione del VPC scade, lo stato per le nuove implementazioni dei Pod di Windows rimane `ContainerCreating`.

Per risolvere il problema se disponi del supporto Windows legacy sul piano dati, consulta Rinnovo del certificato webhook di ammissione VPC. Se le versioni del cluster e della piattaforma sono successive a una versione elencata in Prerequisiti del supporto Windows, ti consigliamo di rimuovere il supporto Windows legacy dal piano dati e abilitarlo per il tuo piano di controllo. Una volta fatto, non devi gestire il certificato webhook. Per ulteriori informazioni, consulta Abilitazione del supporto di Windows per il cluster Amazon EKS.

Prima di poter aggiornare il piano di controllo i gruppi di nodi devono corrispondere alla versione di Kubernetes

Prima di aggiornare il piano di controllo a una nuova versione di Kubernetes, assicurati che la versione secondaria dei nodi gestiti e Fargate nel cluster sia la stessa della versione corrente del piano di controllo. L'API `update-cluster-version` di Amazon EKS rifiuta le richieste fino a quando non si aggiornano tutti i nodi gestiti Amazon EKS alla versione corrente del cluster. Amazon EKS fornisce API per aggiornare i nodi gestiti. Per informazioni sull'aggiornamento delle versioni di Kubernetes del gruppo di nodi gestiti, consulta la sezione Aggiornamento di un gruppo di nodi gestiti. Per aggiornare la versione di un nodo Fargate, elimina il pod rappresentato dal nodo e implementa nuovamente il pod dopo aver aggiornato il piano di controllo. Per ulteriori informazioni, consulta Aggiornamento della versione di Kubernetes del cluster Amazon EKS.

All'avvio di molti nodi, si verificano errori **`Too Many Requests`**

Se si avviano più nodi contemporaneamente, si potrebbe visualizzare il messaggio di errore `Too Many Requests` nei registri di esecuzione dei dati utente di Amazon EC2. Ciò può verificarsi perché il piano di controllo è sovraccarico di chiamate `describeCluster`. Il sovraccarico si traduce in limitazione (della larghezza di banda della rete), nodi che non eseguono lo script bootstrap e nodo che non vengono aggiunti al cluster.

Assicurarti che gli argomenti `--apiserver-endpoint`, `--b64-cluster-ca` e `--dns-cluster-ip` vengano trasferiti allo script di bootstrap del nodo. Quando includi questi argomenti, non è necessario che lo script bootstrap crei una chiamata `describeCluster`. Ciò contribuisce a impedire che il piano di controllo si sovraccarichi. Per ulteriori informazioni, consulta Fornire i dati utente per passare argomenti al file `bootstrap.sh` incluso nell'AMI Linux/Bottlerocket ottimizzata per Amazon EKS.

Risposta agli errori HTTP 401 Autorizzazione negata per le richieste del server API Kubernetes

Questi errori vengono visualizzati se il token dell'account di servizio di un Pod è scaduto in un cluster.

Il server API Kubernetes del cluster Amazon EKS rifiuta le richieste con token più vecchi di 90 giorni. Nelle versioni di Kubernetes precedenti, i token non avevano una scadenza. Di conseguenza, i client che si basano su questi token devono aggiornarli entro un'ora. Per impedire al server API Kubernetes di rifiutare la richiesta a causa di un token non valido, la versione dell'SDK client Kubernetes utilizzata dal carico di lavoro deve essere uguale o successiva alle versioni riportate di seguito:

- Go versione `0.15.7` e successive
- Python versione `12.0.0` e successive
- Java versione `9.0.0` e successive
- JavaScript versione `0.10.3` e successive
- Ramo `master` di Ruby
- Haskell versione `0.3.0.0`
- C# versione `7.0.5` e successive

Puoi identificare tutti i Pods esistenti nel cluster che utilizzano token obsoleti. Per ulteriori informazioni, consulta Account del servizio Kubernetes.

La versione della piattaforma Amazon EKS è più avanti di due versioni rispetto all'attuale versione della piattaforma

Questo può accadere quando Amazon EKS non è in grado di aggiornare automaticamente la versione della piattaforma del i cluster. Sebbene ci siano molte cause, di seguito riportiamo quelle più comuni. Se uno di questi problemi si applica al tuo cluster, potrebbe ancora funzionare, ma la sua versione della piattaforma non verrà aggiornata da Amazon EKS.

Problema

Il ruolo IAM del cluster è stato eliminato: questo ruolo è stato specificato al momento della creazione del cluster. Puoi visualizzare quale ruolo è stato specificato con il seguente comando. Sostituisci *my-cluster* con il nome del tuo cluster.

```
aws eks describe-cluster --name my-cluster --query cluster.roleArn --output text | cut
 -d / -f 2
```

Di seguito viene riportato un output di esempio:

```
eksClusterRole
```

Soluzione

Crea un nuovo ruolo IAM del cluster con lo stesso nome.

Problema

Una sottorete specificata durante la creazione del cluster è stata eliminata: le sottoreti da utilizzare con il cluster sono state specificate durante la creazione del cluster. Puoi visualizzare le sottoreti specificate con il seguente comando. Sostituisci *my-cluster* con il nome del tuo cluster.

```
aws eks describe-cluster --name my-cluster --query cluster.resourcesVpcConfig.subnetIds
```

Di seguito viene riportato un output di esempio:

```
[
"subnet-EXAMPLE1",
"subnet-EXAMPLE2"
]
```

Soluzione

Verifica se gli ID di sottorete esistono nel tuo account.

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
aws ec2 describe-subnets --filters "Name=vpc-id,Values=$vpc_id" --query
 "Subnets[*].SubnetId"
```

Di seguito viene riportato un output di esempio:

```
[
"subnet-EXAMPLE3",
```

```
"subnet-EXAMPLE4"
]
```

Nel caso in cui gli ID di sottorete restituiti nell'output non corrispondano agli ID di sottorete specificati al momento della creazione del cluster, se desideri che Amazon EKS aggiorni il cluster, devi modificare le sottoreti utilizzate dal cluster. Questo perché se hai specificato più di due sottoreti al momento della creazione del cluster, Amazon EKS seleziona in modo casuale le sottoreti specificate in cui creare nuove interfacce di rete elastiche. Queste interfacce di rete consentono al piano di controllo di comunicare con i nodi. Amazon EKS non aggiornerà il cluster se la sottorete selezionata non esiste. Non hai alcun controllo su quale delle sottoreti specificate al momento della creazione del cluster Amazon EKS sceglie per creare una nuova interfaccia di rete.

Quando avvii un aggiornamento della versione di Kubernetes per il cluster, l'aggiornamento può non riuscire per lo stesso motivo.

Problema

Un gruppo di sicurezza specificato durante la creazione del cluster è stato eliminato: se durante la creazione del cluster hai specificato dei gruppi di sicurezza, puoi visualizzarne gli ID con il seguente comando. Sostituisci *my-cluster* con il nome del tuo cluster.

```
aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.securityGroupIds
```

Di seguito viene riportato un output di esempio:

```
[
    "sg-EXAMPLE1"
]
```

Se viene restituito `[]`, allora non sono stati specificati gruppi di sicurezza al momento della creazione del cluster e il problema non è dato dalla mancanza di un gruppo di sicurezza. Se vengono restituiti i gruppi di sicurezza, conferma che i gruppi di sicurezza sono presenti nel tuo account.

Soluzione

Verifica se questi gruppi di sicurezza esistono nel tuo account.

```
vpc_id=$(aws eks describe-cluster --name my-cluster --query
 cluster.resourcesVpcConfig.vpcId --output text)
```

```
aws ec2 describe-security-groups --filters "Name=vpc-id,Values=$vpc_id" --query
 "SecurityGroups[*].GroupId"
```

Di seguito viene riportato un output di esempio:

```
[
"sg-EXAMPLE2"
]
```

Se desideri che Amazon EKS aggiorni il cluster, nel caso in cui gli ID dei gruppi di sicurezza restituiti nell'output non corrispondano agli ID dei gruppi di sicurezza specificati al momento della creazione del cluster, devi modificare i gruppi di sicurezza utilizzati dal cluster. Amazon EKS non aggiornerà un cluster se gli ID dei gruppi di sicurezza specificati al momento della creazione del cluster non esistono.

Quando avvii un aggiornamento della versione di Kubernetes per il cluster, l'aggiornamento può non riuscire per lo stesso motivo.

Altri motivi per cui Amazon EKS non aggiorna la versione della piattaforma del tuo cluster

- Non hai almeno sei (anche se ne consigliamo 16) indirizzi IP disponibili in ciascuna delle sottoreti specificate al momento della creazione del cluster. Se non disponi di un numero sufficiente di indirizzi IP disponibili nella sottorete, dovrai liberare gli indirizzi IP nella sottorete oppure modificare le sottoreti utilizzate dal cluster in modo da utilizzare le sottoreti con un numero sufficiente di indirizzi IP disponibili.
- Hai abilitato la crittografia dei segreti quando hai creato il cluster e la AWS KMS chiave specificata è stata eliminata. Se desideri che Amazon EKS aggiorni il cluster, devi crearne uno nuovo

Domande frequenti sullo stato del cluster e codici di errore con percorsi di risoluzione

Amazon EKS rileva problemi con i cluster EKS e l'infrastruttura del cluster e li archivia in integrità del cluster. Grazie alle informazioni sull'integrità del cluster è possibile rilevare, risolvere e affrontare più rapidamente eventuali problemi. Ciò consente di creare ambienti applicativi più sicuri e up-to-date. Inoltre, è possibile che tu non possa effettuare l'aggiornamento a versioni più recenti di Kubernetes o che Amazon EKS non sia in grado di installare aggiornamenti di sicurezza su un cluster degradato a causa di problemi relativi all'infrastruttura necessaria o alla configurazione del cluster. Amazon EKS può impiegare tre ore per rilevare problemi o verificare che un problema è stato risolto.

L'integrità di un cluster Amazon EKS rappresenta una responsabilità condivisa tra Amazon EKS e i suoi utenti. Sei responsabile dell'infrastruttura indispensabile per i ruoli IAM e le sottoreti Amazon VPC, nonché di altre infrastrutture necessarie, che devono essere fornite anticipatamente. Amazon EKS rileva le modifiche nella configurazione di questa infrastruttura e del cluster.

Per accedere all'integrità del cluster nella console Amazon EKS, è necessario trovare una sezione chiamata Problemi di integrità nella scheda Panoramica della pagina dei dettagli del cluster Amazon EKS. Questi dati saranno disponibili anche richiamando l'azione `DescribeCluster` nell'API EKS, ad esempio dall'interno della AWS Command Line Interface.

Qual è il vantaggio di utilizzare questa funzionalità?

> Otterrai una maggiore visibilità sullo stato del tuo cluster Amazon EKS, diagnosticherai e risolverai rapidamente eventuali problemi, senza dover dedicare tempo al debug o all'apertura di casi di supporto. AWS Ad esempio: hai eliminato accidentalmente una sottorete per il cluster Amazon EKS, Amazon EKS non sarà in grado di creare interfacce e Kubernetes AWS CLI comandi di rete tra account come `kubectl` exec o logs. `kubectl` Queste operazioni non riusciranno e restituiranno l'errore: `Error from server: error dialing backend: remote error: tls: internal error.` Ora sarà visualizzato un problema di integrità di Amazon EKS con il seguente messaggio: `subnet-da60e280 was deleted: could not create network interface`.

In che modo questa funzionalità è correlata o funziona con altri servizi? AWS

> I ruoli IAM e le sottoreti Amazon VPC sono due esempi di elementi dell'infrastrutture indispensabile per i quali l'integrità del cluster può rilevare problemi. Se tali risorse non sono configurate correttamente, questa funzionalità restituirà informazioni dettagliate.

Un cluster con problemi di integrità comporta costi?

> Sì, ogni cluster Amazon EKS viene fatturato ai prezzi standard di Amazon EKS. La funzionalità integrità del cluster è disponibile senza costi aggiuntivi.

Questa funzionalità funziona con i cluster Amazon EKS su AWS Outposts?

> Sì, vengono rilevati problemi di cluster per i cluster EKS nel AWS cloud, inclusi i cluster estesi attivi AWS Outposts e i cluster locali attivi. AWS Outposts L'integrità del cluster non rileva problemi con Amazon EKS Anywhere o Amazon EKS Distro (EKS-D).

Posso ricevere una notifica quando vengono rilevati nuovi problemi?

> No, è necessario controllare la console Amazon EKS o effettuare una chiamata all'API `DescribeCluster` di EKS.

La console mi avvisa in caso di problemi di integrità?

Sì, qualsiasi cluster con problemi di integrità presenterà un banner di avviso nella parte superiore della console.

Le prime due colonne sono quelle necessarie per i valori di risposta dell'API. Il terzo campo dell' ClusterIssueoggetto Health èresourceIds, la cui restituzione dipende dal tipo di problema.

Codice	Messaggio	ResourceIds	Cluster recuperabile?	
SUBNET_NOT_FOUND	Non è stato possibile trovare una o più sottoreti attualmente associate al tuo cluster. Effettua una chiamata all'API update-cluster-config di Amazon EKS per aggiornare le sottoreti.	ID sottorete	Sì	
SECURITY_GROUP_NOT_FOUND	Non è stato possibile trovare uno o più gruppi di sicurezza attualmente associati al tuo cluster. Chiama l' update-cluster-config API Amazon EKS per aggiornare i gruppi di sicurezza	ID gruppo di sicurezza	Sì	
IP_NOT_AVAILABLE	Una o più sottoreti associate al cluster non dispone di indirizzi IP disponibili sufficienti per consentire ad Amazon EKS di eseguire operazioni di gestione dei cluster. Libera indirizzi nelle sottoreti o associa diverse sottoreti al cluster utilizzando l'API Amazon EKS. update-cluster-config	ID sottorete	Sì	
VPC_NOT_FOUND	Non è stato possibile trovare il VPC associato al cluster. È	ID VPC	No	

Codice	Messaggio	ResourceIds	Cluster recuperabile?	
	necessario eliminare e ricreare il cluster.			
ASSUME_ROLE_ACCESS_DENIED	Il tuo cluster non utilizza Amazon EKS service-linked-role. Non è stato possibile assumere il ruolo associato al tuo cluster per eseguire le operazioni di gestione di Amazon EKS richieste. Verifica che il ruolo esista e disponga della policy di attendibilità richiesta.	Ruolo IAM del cluster	Sì	
PERMISSION_ACCESS_DENIED	Il tuo cluster non utilizza Amazon EKS service-linked-role. Il ruolo associato al tuo cluster non concede autorizzazioni sufficienti ad Amazon EKS per eseguire le operazioni di gestione richieste. Verifica le policy associate al ruolo del cluster e controlla se vengono applicate policy di negazione separate.	Ruolo IAM del cluster	Sì	
ASSUME_ROLE_ACCESS_DENIED_USING_SLR	Non potevamo dare per scontato la gestione del cluster Amazon EKS service-linked-role. Verifica che il ruolo esista e disponga della policy di attendibilità richiesta.	Amazon EKS service-linked-role	Sì	

Codice	Messaggio	ResourceIds	Cluster recuperabile?	
PERMISSION_ACCESS_DENIED_USING_SLR	La gestione dei cluster Amazon EKS service-linked-role non concede autorizzazioni sufficienti ad Amazon EKS per eseguire le operazioni di gestione richieste. Verifica le policy associate al ruolo del cluster e controlla se vengono applicate policy di negazione separate.	Amazon EKS service-linked-role	Sì	
OPT_IN_REQUIRED	Il tuo account non dispone di un abbonamento al servizio Amazon EC2. Aggiorna gli abbonamenti del tuo account nella pagina delle impostazioni dell'account.	N/D	Sì	
STS_REGIONAL_ENDPOINT_DISABLED	L'endpoint regionale STS è disabilitato. Abilita l'endpoint per Amazon EKS per eseguire le operazioni di gestione dei cluster richieste.	N/D	Sì	
KMS_KEY_DISABLED	La AWS KMS chiave associata al cluster è disabilitata. Riabilita la chiave per ripristinare il cluster.	Il KMS Key Arn	Sì	
KMS_KEY_NOT_FOUND	Non siamo riusciti a trovare la AWS KMS chiave associata al tuo cluster. È necessario eliminare e ricreare il cluster.	Il KMS Key ARN	No	

Codice	Messaggio	ResourceIds	Cluster recuperabile?	
KMS_GRANT_REVOKED	Le concessioni per la AWS KMS chiave associata al tuo cluster vengono revocate. È necessario eliminare e ricreare il cluster.	Il KMS Key Arn	No	

Amazon EKS Connector

È possibile usare Amazon EKS Connector per registrare e connettere qualsiasi cluster Kubernetes conforme a AWS e visualizzarlo nella console Amazon EKS. Una volta connesso, è possibile visualizzare lo stato, la configurazione e i carichi di lavoro del cluster nella console Amazon EKS. È possibile utilizzare questa funzione per visualizzare i cluster connessi nella console Amazon EKS, ma non è possibile gestirli. Amazon EKS Connector richiede un agente che è un progetto open source su GitHub. Per altri contenuti tecnici, incluse le domande frequenti e la risoluzione dei problemi, consultare Risoluzione dei problemi in Amazon EKS Connector

Amazon EKS Connector connette i seguenti tipi di cluster Kubernetes ad Amazon EKS.

- Cluster Kubernetes on-premise
- Cluster autogestiti in esecuzione su Amazon EC2
- Cluster gestiti da altri provider cloud

Considerazioni su Amazon EKS Connector

Prima di utilizzare Amazon EKS Connector, attenersi a quanto segue:

- Prima di connettere il cluster ad Amazon EKS, è necessario disporre dei privilegi amministrativi per il cluster Kubernetes.
- Il cluster Kubernetes deve avere nodi worker Linux a 64 bit (x86) presenti prima della connessione. I nodi (worker) ARM non sono supportati.
- È necessario disporre di nodi worker nel cluster Kubernetes che abbiano accesso in uscita agli endpoint di Systems Manager `ssm.` e `ssmmessages.`. Per ulteriori informazioni, consulta Endpoint di Systems Manager in Riferimenti generali di AWS.
- Per impostazione predefinita, è possibile connettere fino a 10 cluster per regione. È possibile richiedere un aumento tramite la console Service Quotas. Per ulteriori informazioni, consultare Richiesta di un aumento di quota.
- Solo le API `RegisterCluster`, `ListClusters`, `DescribeCluster` e `DeregisterCluster` di Amazon EKS sono supportate per i cluster Kubernetes esterni.
- È necessario disporre delle autorizzazioni seguenti per registrare un cluster:
  - eks:RegisterCluster

- ssm:CreateActivation
- ssm:DeleteActivation
- iam:PassRole

- È necessario disporre delle autorizzazioni seguenti per annullare la registrazione di un cluster:
  - eks:DeregisterCluster
  - ssm:DeleteActivation
  - ssm:DeregisterManagedInstance

Ruoli IAM richiesti per Amazon EKS Connector

L'utilizzo di Amazon EKS Connector richiede i seguenti due ruoli IAM:

- Il ruolo Amazon EKS Connectore collegato al servizio viene creato al momento della prima registrazione del cluster.
- È indispensabile creare il ruolo IAM dell'agente Amazon EKS Connector. Per informazioni dettagliate, consulta Ruolo IAM di Amazon EKS Connector.

Per abilitare l'autorizzazione alla visualizzazione del cluster e del carico di lavoro per i principali IAM, è necessario applicare il `eks-connector` e i ruoli del cluster Amazon EKS Connector nel cluster. Seguire la procedura riportata in Concessione dell'accesso a un principale IAM per la visualizzazione delle risorse Kubernetes in un cluster.

Connessione di un cluster esterno

È possibile collegare un cluster Kubernetes esterno ad Amazon EKS utilizzando più metodi, come illustrato nel seguente processo. Questo processo prevede due passaggi: registrare il cluster con Amazon EKS e installare l'agente `eks-connector` nel cluster.

> **Important**
>
> È necessario completare il secondo passaggio entro 3 giorni dal completamento del primo passaggio, ossia prima che la registrazione scada.

Metodi di connessione

Non tutti i metodi di installazione dell'agente possono essere utilizzati dopo ciascuno dei metodi di registrazione del cluster. Nella tabella seguente sono elencati tutti i metodi di registrazione e i metodi di installazione dell'agente che è possibile utilizzare.

Fase	Metodi		
Registrazione del cluster	AWS Management Console	AWS Command Line Interface	`eksctl`
Installare l'agente	Helm, manifesti YAML	Helm, manifesti YAML	Manifesti YAML

Prerequisiti

- Assicurati di avere creato il ruolo agente del connettore Amazon EKS. Seguire la procedura riportata in Creazione del ruolo agente di Amazon EKS Connector.
- È necessario disporre delle autorizzazioni seguenti per registrare un cluster:
  - `eks:RegisterCluster`
  - `ssm:CreateActivation`
  - `ssm:DeleteActivation`
  - `iam:PassRole`

Fase 1: Registrazione del cluster

AWS CLI

Prerequisiti

- La AWS CLI deve essere installata. Per installarla o aggiornarla, consultare Installazione della AWS CLI.

Registrazione del cluster con la AWS CLI

- Per la configurazione del plug-in, specificare il ruolo IAM dell'agente Amazon EKS Connector. Per ulteriori informazioni, consulta Ruoli IAM richiesti per Amazon EKS Connector.

```
aws eks register-cluster \
     --name my-first-registered-cluster \
     --connector-config roleArn=arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole,provider="OTHER" \
     --region aws-region
```

Di seguito viene riportato un output di esempio:

```
{
    "cluster": {
        "name": "my-first-registered-cluster",
        "arn": "arn:aws:eks:region:111122223333:cluster/my-first-registered-
cluster",
        "createdAt": 1627669203.531,
        "ConnectorConfig": {
            "activationId": "xxxxxxxxACTIVATION_IDxxxxxxxx",
            "activationCode": "xxxxxxxxACTIVATION_CODExxxxxxxx",
            "activationExpiry": 1627672543.0,
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/
AmazonEKSConnectorAgentRole"
        },
        "status": "CREATING"
    }
}
```

I valori `aws-region`, `activationId`, e `activationCode` verranno utilizzati nel prossimo passaggio.

AWS Management Console

Per registrare il cluster Kubernetes con la console.

1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli Add cluster (Aggiungi cluster) e seleziona Register (Registra) per visualizzare la pagina di configurazione.
3. Nella pagina Configura cluster compilare i campi riportati di seguito:

- Nome: un nome univoco per il cluster.
- Provider: scegli per visualizzare l'elenco a discesa dei provider del cluster Kubernetes. Se non si conosce il provider specifico, selezionare Altro.
- Ruolo plug-in EKS: selezionare il ruolo da utilizzare per la connessione del cluster.

4. Selezionare Registra cluster.
5. Viene visualizzata la pagina Panoramica del cluster. Se desideri utilizzare il grafico Helm, copia il comando `helm install` e prosegui con il passaggio successivo. Se desideri utilizzare il manifesto YAML, scegli Scarica file YAML per scaricare il file manifesto sull'unità locale.

> **Important**
>
> - Questa è l'unica occasione per copiare il comando `helm install` o eseguire il download di questo file. Non uscire da questa pagina, poiché il collegamento non sarà accessibile e sarà necessario annullare la registrazione del cluster e riavviare i passaggi dall'inizio.
> - Il comando o il file manifesto può essere utilizzato una sola volta per il cluster registrato. Se elimini risorse dal cluster Kubernetes, dovrai registrare nuovamente il cluster e ottenere un nuovo file manifesto.

Passa alla fase successiva per applicare il file manifesto al cluster Kubernetes.

eksctl

Prerequisiti

- Deve essere installato `eksctl` versione `0.68` o successiva. Per installarlo o aggiornarlo, consulta la pagina Guida introduttiva ad Amazon EKS: `eksctl`.

Registrazione del cluster con **`eksctl`**

1. Registra il cluster fornendo un nome, un provider e una regione.

```
eksctl register cluster --name my-cluster --provider my-provider --
region region-code
```

Output di esempio:

```
2021-08-19 13:47:26 [#]  creating IAM role "eksctl-20210819194112186040"
2021-08-19 13:47:26 [#]  registered cluster "<name>" successfully
2021-08-19 13:47:26 [#]  wrote file eks-connector.yaml to <current directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-clusterrole.yaml to <current
 directory>
2021-08-19 13:47:26 [#]  wrote file eks-connector-console-dashboard-full-access-
group.yaml to <current directory>
2021-08-19 13:47:26 [!]  note: "eks-connector-clusterrole.yaml" and "eks-
connector-console-dashboard-full-access-group.yaml" give full EKS Console access
 to IAM identity "<aws-arn>", edit if required; read https://eksctl.io/usage/
eks-connector for more info
2021-08-19 13:47:26 [#]  run `kubectl apply -f eks-connector.yaml,eks-connector-
clusterrole.yaml,eks-connector-console-dashboard-full-access-group.yaml` before
 expiry> to connect the cluster
```

In questo modo vengono creati file sul computer locale. Questi file devono essere applicati al cluster esterno entro 3 giorni; in caso contrario, la registrazione scade.

2. In un terminale in grado di accedere al cluster, applica il file `eks-connector-binding.yaml`:

```
kubectl apply -f eks-connector-binding.yaml
```

Passaggio 2: installazione dell'agente **`eks-connector`**

Helm chart

1. Se nel passaggio precedente hai utilizzato la AWS CLI, sostituisci i valori di `ACTIVATION_CODE` e `ACTIVATION_ID` nel seguente comando rispettivamente con i valori `activationId` e `activationCode`. Sostituisci `aws-region` con la Regione AWS utilizzata nella fase precedente. Dopodiché, esegui il comando seguente per installare l'agente `eks-connector` nel cluster in fase di registrazione:

```
$ helm install eks-connector \
  --namespace eks-connector \
  oci://public.ecr.aws/eks-connector/eks-connector-chart \
  --set eks.activationCode=ACTIVATION_CODE \
  --set eks.activationId=ACTIVATION_ID \
  --set eks.agentRegion=aws-region
```

Se nel passaggio precedente hai utilizzato la AWS Management Console, utilizza il comando copiato in tale passaggio inserendovi questi valori.

2. Verifica l'integrità dell'implementazione del `eks-connector` installato e attendi che lo stato del cluster registrato in Amazon EKS diventi `ACTIVE`.

YAML manifest

Completa la connessione applicando il file manifesto di Amazon EKS Connector al tuo cluster Kubernetes. A tale scopo, è necessario utilizzare i metodi descritti in precedenza. Se il manifesto non viene applicato entro tre giorni, la registrazione di Amazon EKS Connector scade. Se la connessione al cluster scade, la registrazione del cluster deve essere annullata prima di poterlo connettere nuovamente.

1. Eseguire il download del file YAML di Amazon EKS Connector.

```
curl -O https://amazon-eks.s3.us-west-2.amazonaws.com/eks-connector/manifests/
eks-connector/latest/eks-connector.yaml
```

2. Modifica il file YAML del connettore Amazon EKS per sostituire tutti i riferimenti di `%AWS_REGION%`, `%EKS_ACTIVATION_ID%` e `%EKS_ACTIVATION_CODE%` con i valori `aws-region`, `activationId`, e `activationCode` dell'output del passaggio precedente.

   Il comando di esempio seguente può sostituire questi valori.

```
sed -i "s~%AWS_REGION%~$aws-region~g; s~%EKS_ACTIVATION_ID
%~$EKS_ACTIVATION_ID~g; s~%EKS_ACTIVATION_CODE%~$(echo -n $EKS_ACTIVATION_CODE |
 base64)~g" eks-connector.yaml
```

> **Important**
>
> Assicurarsi che il codice di attivazione sia in formato base64.

3. In un terminale dotato di accesso al cluster, è possibile applicare il file manifesto aggiornato eseguendo il seguente comando:

```
kubectl apply -f eks-connector.yaml
```

4. Dopo aver applicato il manifesto di Amazon EKS Connector e i file YAML di associazione al ruolo al cluster Kubernetes, verifica che il cluster sia connesso.

```
aws eks describe-cluster \
     --name "my-first-registered-cluster" \
     --region AWS_REGION
```

   L'output deve includere `status=ACTIVE`.

5. (Facoltativo) Assegna tag al cluster. Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Amazon EKS.

Passaggi successivi

In caso di problemi con questi passaggi, consulta la pagina Risoluzione dei problemi in Amazon EKS Connector.

Per concedere a principali IAM aggiuntivi l'accesso alla console Amazon EKS al fine di visualizzare le risorse Kubernetes in un cluster connesso, consulta Concessione dell'accesso a un principale IAM per la visualizzazione delle risorse Kubernetes in un cluster.

Concessione dell'accesso a un principale IAM per la visualizzazione delle risorse Kubernetes in un cluster

Concedi ai principali IAM l'accesso alla console Amazon EKS per visualizzare informazioni sulle risorse Kubernetes in esecuzione sul cluster connesso.

Prerequisiti

Il principale IAM utilizzato per accedere alla AWS Management Console deve soddisfare i seguenti requisiti:

- Deve disporre dell'autorizzazione IAM `eks:AccessKubernetesApi`.
- L'account di servizio Amazon EKS Connector dovrebbe essere in grado di rappresentare il principale IAM nel cluster. Ciò consente ad Amazon EKS Connector di mappare il principale IAM a un utente Kubernetes.

Per creare e applicare il ruolo del cluster Amazon EKS Connector

1. Eseguire il download del modello di ruolo del cluster `eks-connector`.

   ```
   curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-clusterrole.yaml
   ```

2. Modificare il file YAML del modello di ruolo del cluster. Sostituire i riferimenti di `%IAM_ARN%` con il nome della risorsa Amazon (ARN) del principale IAM.
3. Applica il file YAML del ruolo del cluster di Amazon EKS Connector al cluster Kubernetes.

   ```
   kubectl apply -f eks-connector-clusterrole.yaml
   ```

Affinché un principale IAM possa visualizzare le risorse Kubernetes nella console Amazon EKS, questo deve essere associato a un `role` o `clusterrole` Kubernetes con le autorizzazioni necessarie per leggere le risorse. Per ulteriori informazioni, consulta Utilizzo dell'autorizzazione RBAC nella documentazione di Kubernetes.

Configurazione di un principale IAM per accedere al cluster connesso

1. È possibile eseguire il download di questi file manifesto di esempio per creare, rispettivamente, un `clusterrole` e un'`clusterrolebinding` oppure un `role` e un'`rolebinding`:

   Visualizzazione delle risorse Kubernetes in tutti gli spazi dei nomi

   Il ruolo del cluster `eks-connector-console-dashboard-full-access-clusterrole` consente di accedere a tutti gli spazi dei nomi e alle risorse visualizzabili nella console. É possibile modificare il nome di `role`, `clusterrole` e il loro legame corrispondente prima

di applicarlo al cluster. Utilizzare il seguente comando per eseguire il download di un file di esempio.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-console-dashboard-full-access-group.yaml
```

Visualizzazione delle risorse Kubernetes in uno spazio dei nomi specifico

Lo spazio dei nomi in questo file è `default`, quindi se desideri specificare uno spazio dei nomi diverso, modifica il file prima di applicarlo al cluster. Utilizza il comando seguente per avviare il download di un file di esempio.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/eks-connector/manifests/eks-connector-console-roles/eks-connector-console-dashboard-restricted-access-group.yaml
```

2. Modificare il file YAML con accesso completo o con accesso limitato per sostituire i riferimenti di `%IAM_ARN%` con il nome della risorsa Amazon (ARN) del principale IAM.
3. Applica i file YAML ad accesso completo o limitato al cluster Kubernetes. Sostituire il valore del file YAML con il proprio valore.

```
kubectl apply -f eks-connector-console-dashboard-full-access-group.yaml
```

Per visualizzare le risorse Kubernetes nel cluster connesso, consulta Visualizzazione delle risorse Kubernetes. I dati relativi ad alcuni tipi di risorse nella scheda Risorse non sono disponibili per i cluster collegati.

Annullamento della registrazione di un cluster

Al termine dell'utilizzo di un cluster connesso, è possibile annullarne la registrazione. Una volta annullata la registrazione, il cluster non sarà più visibile nella console Amazon EKS.

È necessario disporre delle autorizzazioni seguenti per chiamare l'API di deregisterCluster:

- `eks:DeregisterCluster`
- `ssm:DeleteActivation`
- `ssm:DeregisterManagedInstance`

Questo processo prevede due passaggi: annullare la registrazione del cluster con Amazon EKS e disinstallare l'agente eks-connector nel cluster.

Per annullare la registrazione del cluster Kubernetes

AWS CLI

Prerequisiti

- La AWS CLI deve essere installata. Per installarla o aggiornarla, consultare Installazione della AWS CLI.
- Assicurarsi che sia stato creato il ruolo agente Amazon EKS Connector. .

Annullare la registrazione del cluster connesso.

```
aws eks deregister-cluster \
    --name my-cluster \
    --region region-code
```

AWS Management Console

1. Aprire la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
2. Scegli Cluster.
3. Nella pagina Cluster, selezionare il cluster connesso e selezionare Annulla registrazione.
4. Confermare che si desidera annullare la registrazione del cluster.

eksctl

Prerequisiti

- Deve essere installato `eksctl` versione `0.68` o successiva. Per installarlo o aggiornarlo, consulta la pagina Guida introduttiva ad Amazon EKS: `eksctl`.
- Assicurarsi che sia stato creato il ruolo agente Amazon EKS Connector. .

Annullamento della registrazione del cluster con **eksctl**

- Per la configurazione del plug-in, specificare il ruolo IAM dell'agente Amazon EKS Connector. Per ulteriori informazioni, consulta Ruoli IAM richiesti per Amazon EKS Connector.

```
eksctl deregister cluster --name my-cluster
```

Per eliminare le risorse nel cluster Kubernetes

Helm

- Per disinstallare l'agente, esegui il comando seguente.

```
helm -n eks-connector uninstall eks-connector
```

YAML manifest

1. Elimina il file YAML di Amazon EKS Connector dal cluster Kubernetes.

```
kubectl delete -f eks-connector.yaml
```

2. Se è stato creato un `clusterrole` o `clusterrolebindings` per fornire accesso al cluster a principali IAM aggiuntivi, assicurati di eliminarli dal cluster Kubernetes.

Risoluzione dei problemi in Amazon EKS Connector

In questo argomento sono descritti alcuni degli errori più comuni che si possono verificare durante l'utilizzo del connettore Amazon EKS e sono riportate le istruzioni per risolverli in maniera temporanea o definitiva.

Risoluzione dei problemi di base

In questa sezione sono descritti i passaggi per la diagnosi di un problema non chiaro.

Verifica dello stato di Amazon EKS Connector

Controlla lo stato del connettore Amazon EKS.

```
kubectl get pods -n eks-connector
```

Ispezione dei log del connettore Amazon EKS

Il Pod del connettore Amazon EKS è composto da tre container. Per recuperare i registri completi per tutti questi container in modo da poterli ispezionare, eseguire i seguenti comandi:

- `connector-init`

```
kubectl logs eks-connector-0 --container connector-init -n eks-connector
kubectl logs eks-connector-1 --container connector-init -n eks-connector
```

- `connector-proxy`

```
kubectl logs eks-connector-0 --container connector-proxy -n eks-connector
kubectl logs eks-connector-1 --container connector-proxy -n eks-connector
```

- `connector-agent`

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector -- cat /
var/log/amazon/ssm/amazon-ssm-agent.log
kubectl exec eks-connector-1 --container connector-agent -n eks-connector -- cat /
var/log/amazon/ssm/amazon-ssm-agent.log
```

Ottenimento del nome effettivo del cluster

I cluster Amazon EKS sono identificati in modo univoco da `clusterName` all'interno di un singolo account AWS e Regione AWS. Se esistono più cluster connessi in Amazon EKS, puoi verificare con quale cluster di Amazon EKS è registrato il cluster Kubernetes corrente. Per fare ciò, inserisci quanto segue per individuare il `clusterName` del cluster corrente.

```
kubectl exec eks-connector-0 --container connector-agent -n eks-connector \
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
 | sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
kubectl exec eks-connector-1 --container connector-agent -n eks-connector \
  -- cat /var/log/amazon/ssm/amazon-ssm-agent.log | grep -m1 -oE "eks_c:[a-zA-Z0-9_-]+"
 | sed -E "s/^.*eks_c:([a-zA-Z0-9_-]+)_[a-zA-Z0-9]+.*$/\1/"
```

Comandi vari

I seguenti comandi sono utili per recuperare le informazioni necessarie per risolvere i problemi.

- Utilizza il seguente comando per raccogliere immagini utilizzate dai Pods in Amazon EKS Connector.

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.containers[*].image}"
 | tr -s '[[:space:]]' '\n'
```

- Usa il comando seguente per determinare i nomi dei nodi su cui è in esecuzione il connettore Amazon EKS.

```
kubectl get pods -n eks-connector -o jsonpath="{.items[*].spec.nodeName}" | tr -s
 '[[:space:]]' '\n'
```

- Per ottenere le versioni client e server Kubernetes esegui il comando riportato.

```
kubectl version
```

- Usa il comando seguente per ottenere informazioni sui tuoi nodi.

```
kubectl get nodes -o wide --show-labels
```

Problema di Helm: 403 Forbidden

Se hai riscontrato il seguente errore durante l'esecuzione dei comandi di installazione di helm:

```
Error: INSTALLATION FAILED: unexpected status from HEAD request to https://
public.ecr.aws/v2/eks-connector/eks-connector-chart/manifests/0.0.6: 403 Forbidden
```

Puoi eseguire la riga seguente per risolvere il problema:

```
docker logout public.ecr.aws
```

Errore della console: il cluster è bloccato nello stato Pending (In sospeso)

Se il cluster rimane bloccato nello `Pending` stato della console Amazon EKS dopo la registrazione, è possibile che Amazon EKS Connector non abbia AWS ancora collegato correttamente il cluster.

Per un cluster registrato, lo stato `Pending` implica che la connessione non è stata stabilita correttamente. Per risolvere il problema, assicurati di aver applicato il manifesto al cluster Kubernetes di destinazione. Se l'hai applicato al cluster ma il cluster rimane ancora nello stato `Pending`, lo statefulset `eks-connector` potrebbe non essere integro. Per risolvere questo problema, consulta I Pods di Amazon EKS Connector stanno eseguendo un ciclo di arresto anomalo in questo argomento.

Errore della console: **`User “system:serviceaccount:eks-connector:eks-connector” can't impersonate resource “users” in API group “”`** in ambito cluster

Il connettore Amazon EKS utilizza la rappresentazione dell'utente di Kubernetes per agire per conto dei principali IAM dalla AWS Management Console. A ogni principale che accede all'KubernetesAPI dall'account del AWS `eks-connector` servizio deve essere concessa l'autorizzazione a impersonare l'Kubernetesutente corrispondente con un ARN IAM come nome utente. Kubernetes Negli esempi seguenti, l'ARN IAM viene mappato a un utente Kubernetes.

- L'utente IAM *`john`* dall' AWS account *`111122223333`* viene mappato su un utente. Kubernetes Le best practice IAM consigliano di concedere le autorizzazioni ai ruoli anziché agli utenti.

  ```
  arn:aws:iam::111122223333:user/john
  ```

- Il ruolo IAM *`admin`* proveniente dall' AWS account *`111122223333`* è mappato a un Kubernetes utente:

  ```
  arn:aws:iam::111122223333:role/admin
  ```

  Il risultato è un ARN del ruolo IAM, anziché l'ARN della sessione AWS STS .

Per le istruzioni per configurare `ClusterRole` e `ClusterRoleBinding` per concedere il privilegio dell'account di servizio `eks-connector` per impersonare l'utente mappato, consulta Concessione dell'accesso a un principale IAM per la visualizzazione delle risorse Kubernetes in un cluster. Accertati che nel modello `%IAM_ARN%` sia sostituito con l'ARN IAM del principale IAM della AWS Management Console .

Errore della console: **[...] is forbidden: User [...] cannot list resource "[...] in API group"** in ambito cluster

Consideriamo il seguente problema. Amazon EKS Connector ha impersonato con successo il principale AWS Management Console IAM richiedente nel cluster di destinazione. Kubernetes Tuttavia, il principale impersonato non dispone dell'autorizzazione RBAC per le operazioni API di Kubernetes.

Per risolvere questo problema, esistono due metodi per concedere le autorizzazioni ad altri utenti. Se in precedenza hai installato eks-connector tramite un grafico Helm, puoi concedere facilmente l'accesso agli utenti eseguendo il seguente comando. Sostituisci i valori `userARN1` e `userARN2` con un elenco degli ARN dei ruoli IAM a cui consentire l'accesso per visualizzare le risorse Kubernetes:

```
helm upgrade eks-connector oci://public.ecr.aws/eks-connector/eks-connector-chart \
    --reuse-values \
    --set 'authentication.allowedUserARNs={userARN1,userARN2}'
```

In alternativa, in qualità di amministratore del cluster, concedi il livello appropriato di privilegi RBAC ai singoli utenti Kubernetes. Per maggiori informazioni ed esempi, consulta Concessione dell'accesso a un principale IAM per la visualizzazione delle risorse Kubernetes in un cluster.

Errore della console: Amazon EKS non è in grado di comunicare con il server API del cluster Kubernetes. Per una connessione corretta, il cluster deve trovarsi nello stato ACTIVE (ATTIVO). Riprova tra qualche minuto.

Se il servizio Amazon EKS non può comunicare con il connettore Amazon EKS nel cluster di destinazione, il motivo potrebbe essere uno dei seguenti:

- Il connettore Amazon EKS nel cluster di destinazione non è integro.
- Connettività scadente o connessione interrotta tra il cluster di destinazione e la Regione AWS.

Per risolvere questo problema, controlla Amazon EKS Connector logs (Log del connettore Amazon EKS). Se non visualizzi un errore per il connettore Amazon EKS, ritenta la connessione dopo qualche minuto. Se riscontri regolarmente una connettività ad alta latenza o intermittente per il cluster di destinazione, prendi in considerazione la possibilità di registrare nuovamente il cluster su uno più vicino a te. Regione AWS

I Pods di Amazon EKS Connector stanno eseguendo un ciclo di arresto anomalo

Esistono molti motivi che possono causare il passaggio di un Pod del connettore Amazon EKS nello stato `CrashLoopBackOff`. Questo problema riguarda probabilmente il container `connector-init`. Controlla lo stato Pod del connettore Amazon EKS.

```
kubectl get pods -n eks-connector
```

Di seguito viene riportato un output di esempio:

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:CrashLoopBackOff   1          7s
```

Se l'output è simile a quello precedente, consulta la pagina Ispezione dei log del connettore Amazon EKS per risolvere il problema.

`Failed to initiate eks-connector: InvalidActivation`

Quando avvii il connettore Amazon EKS per la prima volta, il connettore registra un `activationId` e un `activationCode` con Amazon Web Services. La registrazione potrebbe non riuscire, causando un arresto anomalo del container `connector-init` con un errore simile al seguente.

```
F1116 20:30:47.261469       1 init.go:43] failed to initiate eks-connector:
 InvalidActivation:
```

Per risolvere questo problema, prendere in considerazione le seguenti cause e le correzioni consigliate:

- La registrazione potrebbe non essere riuscita perché nel file manifesto mancano `activationId` e `activationCode`. In questo caso, accertati che siano i valori corretti restituiti dall'operazione API `RegisterCluster` e che `activationCode` sia presente nel file manifesto. `activationCode` viene aggiunto ai segreti Kubernetes, per cui deve essere codificato in `base64`. Per ulteriori informazioni, consulta Fase 1: Registrazione del cluster.
- La registrazione potrebbe non essere riuscita perché l'attivazione è scaduta. Ciò accade perché, per motivi di sicurezza, devi attivare il connettore Amazon EKS entro tre giorni dalla registrazione del cluster. Per risolvere questo problema, accertati che il manifesto del connettore Amazon

EKS sia applicato al cluster Kubernetes di destinazione prima della data e ora di scadenza. Per verificare la data di scadenza dell'attivazione, richiama l'operazione API `DescribeCluster`.

```
aws eks describe-cluster --name my-cluster
```

Nella risposta di esempio seguente, la data e l'ora di scadenza vengono registrate come `2021-11-12T22:28:51.101000-08:00`.

```
{
    "cluster": {
        "name": "my-cluster",
        "arn": "arn:aws:eks:region:111122223333:cluster/my-cluster",
        "createdAt": "2021-11-09T22:28:51.449000-08:00",
        "status": "FAILED",
        "tags": {
        },
        "connectorConfig": {
            "activationId": "00000000-0000-0000-0000-000000000000",
            "activationExpiry": "2021-11-12T22:28:51.101000-08:00",
            "provider": "OTHER",
            "roleArn": "arn:aws:iam::111122223333:role/my-connector-role"
        }
    }
}
```

Se `activationExpiry` è passato, annulla la registrazione del cluster e registralo di nuovo. In questo modo si genera una nuova attivazione.

Nel nodo del cluster manca la connettività in uscita

Per il corretto funzionamento, il connettore Amazon EKS richiede la connettività in uscita a diversi endpoint AWS . Non è possibile connettere un cluster privato senza connettività in uscita a una Regione AWS di destinazione. Per risolvere il problema, è necessario aggiungere la connettività in uscita necessaria. Per informazioni sui requisiti del connettore, consultare Considerazioni su Amazon EKS Connector.

I Pods del connettore Amazon EKS sono in stato `ImagePullBackOff`

Se esegui il comando `get pods` e i Pods sono in stato `ImagePullBackOff`, non possono funzionare correttamente. Se i Pods del connettore Amazon EKS sono in stato `ImagePullBackOff`, non possono funzionare correttamente. Controlla lo stato dei Pods del connettore Amazon EKS.

```
kubectl get pods -n eks-connector
```

Di seguito viene riportato un output di esempio:

```
NAME              READY   STATUS                  RESTARTS   AGE
eks-connector-0   0/2     Init:ImagePullBackOff   0          4s
```

Il file manifesto predefinito del connettore Amazon EKS fa riferimento alle immagini di Amazon ECR Public Gallery. È possibile che il cluster Kubernetes di destinazione non possa estrarre immagini da Amazon ECR Public Gallery. Risolvi il problema di estrazione delle immagini di Amazon ECR Public Gallery o valuta il mirroring delle immagini nel registro del container privato di tua scelta.

Domande frequenti

D: Come funziona la tecnologia alla base del connettore Amazon EKS?

R: Il connettore Amazon EKS è basato sull'agente AWS Systems Manager (System Manager). Il connettore Amazon EKS viene eseguito come `StatefulSet` nel tuo cluster Kubernetes. Stabilisce una connessione e applica un proxy alla comunicazione tra il server API del cluster e Amazon Web Services. Ciò è necessario per visualizzare i dati del cluster nella console Amazon EKS fino a quando non si disconnette il cluster da AWS. L'agente Systems Manager è un progetto open source. Per ulteriori informazioni su questo progetto, consulta la pagina del progetto su GitHub.

D: Dispongo di un cluster Kubernetes on-premise a cui desidero connettermi. Devo aprire le porte firewall per connetterlo?

R: No, non è necessario aprire alcuna porta firewall. Il cluster Kubernetes richiede solo la connessione in uscita a Regioni AWS. I servizi AWS non accedono mai alle risorse nella rete on-premise. Amazon EKS Connector viene eseguito sul cluster e avvia la connessione a AWS. Al termine della registrazione del cluster, AWS emette comandi su Amazon EKS Connector solo dopo aver avviato un'azione dalla console Amazon EKS che richiede informazioni dal server API Kubernetessul cluster.

D: Quali dati vengono inviati dal cluster a AWS tramite Amazon EKS Connector?

R: Amazon EKS Connector invia le informazioni tecniche necessarie per la registrazione del cluster su AWS. Invia inoltre metadati di cluster e carichi di lavoro per le funzionalità della console Amazon EKS richieste dai clienti. Il connettore Amazon EKS raccoglie o invia questi dati solo se avvii un'azione dalla console Amazon EKS o dall'API Amazon EKS che richiede l'invio dei dati a AWS. Per impostazione predefinita, oltre al numero di versione di Kubernetes, AWS non memorizza dati. Memorizza i dati solo se lo autorizzi.

D: È possibile collegare un cluster al di fuori di una Regione AWS?

R: Sì, puoi connettere un cluster da qualsiasi posizione ad Amazon EKS. Inoltre, il servizio Amazon EKS può essere collocato in qualsiasi Regione AWS commerciale pubblica AWS. Funziona con una connessione di rete valida dal cluster alla Regione AWS di destinazione. È preferibile scegliere la Regione AWS più vicina alla posizione del tuo cluster per ottimizzare le prestazioni dell'interfaccia utente. Ad esempio, se hai un cluster in esecuzione a Tokyo, connetti il cluster alla Regione AWS di Tokyo (cioè la `ap-northeast-1` Regione AWS) per assicurare una bassa latenza. Puoi collegare un cluster da qualunque posizione ad Amazon EKS in una delle Regioni AWS commerciali pubbliche, tranne le Regioni AWS Cina o GovCloud.

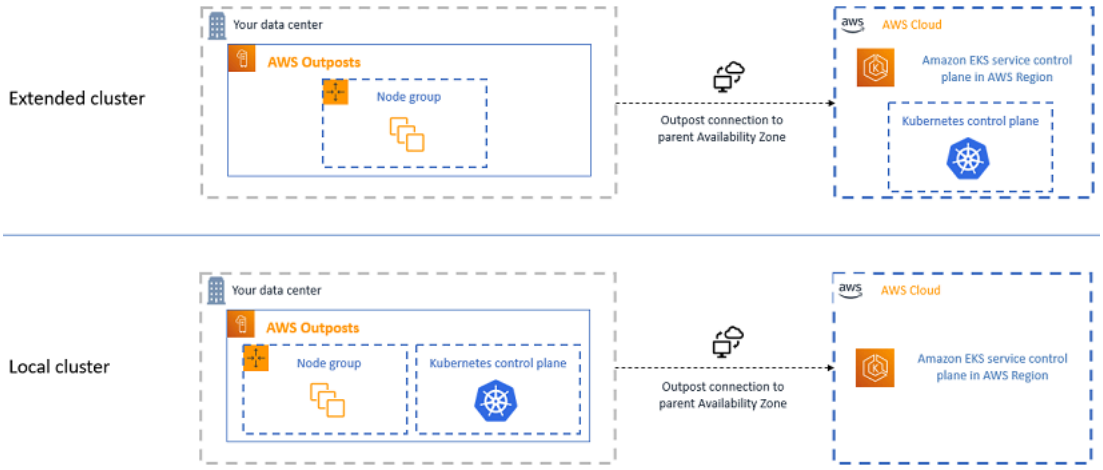
Amazon EKS su AWS Outposts

Puoi utilizzare Amazon EKS per eseguire applicazioni Kubernetes on-premise su AWS Outposts. Puoi implementare Amazon EKS su Outposts nei seguenti modi:

- Cluster estesi: esegui il piano di controllo (control-plane) Kubernetes in una Regione AWS e i nodi sull'Outpost.
- Cluster locali: esegui il piano di controllo Kubernetes e i nodi sull'Outpost.

Per entrambe le opzioni di implementazione, il piano di controllo Kubernetes è completamente gestito da AWS. Puoi utilizzare le stesse API, gli stessi strumenti e le stesse console di Amazon EKS che utilizzi nel cloud per creare ed eseguire Amazon EKS su Outposts.

Il diagramma seguente illustra queste opzioni di implementazione.



Quando utilizzare ciascuna opzione di implementazione

Sia i cluster locali sia quelli estesi sono opzioni di implementazione generiche e possono essere utilizzati per una vasta gamma di applicazioni.

Con i cluster locali puoi eseguire l'intero cluster Amazon EKS in locale su Outposts. Questa opzione consente di ridurre il rischio di tempi di inattività delle applicazioni che possono derivare da disconnessioni temporanee del cloud dalla rete. Queste disconnessioni di rete possono essere causate da interruzioni della fibra o da eventi meteorologici. Poiché l'intero cluster Amazon EKS viene eseguito in locale su Outposts, le applicazioni rimangono disponibili. Durante le disconnessioni del cloud dalla rete è possibile eseguire operazioni del cluster. Per ulteriori informazioni, consulta

Preparazione alle disconnessioni dalla rete. Se ti preoccupa la qualità della connessione di rete dai tuoi Outpost alla Regione AWS principale e hai bisogno di una disponibilità elevata durante le disconnessioni dalla rete, ti consigliamo di utilizzare l'opzione di implementazione del cluster locale.

I cluster estesi consentono di risparmiare capacità sul tuo Outpost perché il piano di controllo Kubernetes viene eseguito nella Regione AWS principale. Questa opzione può essere più adatta se sei in grado di investire in una connettività di rete affidabile e ridondante dal tuo Outpost alla Regione AWS. La qualità della connessione di rete è fondamentale per questa opzione. Il modo in cui Kubernetes gestisce le disconnessioni di rete tra il piano di controllo Kubernetes e i nodi potrebbe comportare tempi di inattività delle applicazioni. Per ulteriori informazioni sul funzionamento di Kubernetes, consulta la sezione Scheduling, Preemption, and Eviction (Pianificazione, precedenza ed espulsione) nella documentazione di Kubernetes.

Confronto tra le opzioni di implementazione

La tabella seguente riporta le differenze tra le due opzioni.

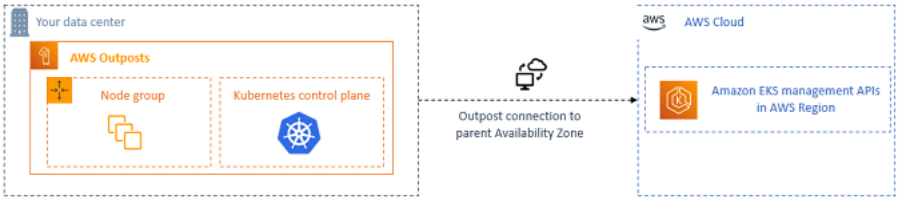
Funzionalità	Cluster esteso	Cluster locale
Posizione del piano di controllo Kubernetes	Regione AWS	Outpost
Account del piano di controllo Kubernetes	Account AWS	Il tuo account
Disponibilità regionale	Consulta la sezione Endpoint del servizio	Stati Uniti orientali (Ohio), Stati Uniti orientali (Virginia settentrionale), Stati Uniti occidentali (California settentrionale), Stati Uniti occidentali (Oregon), Asia Pacifico (Seoul), Asia Pacifico (Singapore), Asia Pacifico (Sydney), Asia Pacifico (Tokyo), Canada (Centrale), Europa (Francoforte), Europa (Irlanda), Europa (Londra),

Funzionalità	Cluster esteso	Cluster locale
		Medio Oriente (Bahrein) e Sud America (San Paolo)
Versioni secondarie di Kubernetes	Versioni supportate da Amazon EKS.	Versioni supportate da Amazon EKS.
Versioni della piattaforma	Consulta la sezione Versioni della piattaforma Amazon EKS	Consulta la sezione Versioni della piattaforma del cluster locale Amazon EKS
Fattori di forma dell'Outpost	Rack dell'Outpost	Rack dell'Outpost
Interfacce utente	AWS Management Console, AWS CLI, API Amazon EKS, `eksctl`, AWS CloudFormation e Terraform	AWS Management Console, AWS CLI, API Amazon EKS, `eksctl`, AWS CloudFormation e Terraform
Policy gestite	AmazonEKSClusterPolicy e AmazonEKSServiceRolePolicy	AmazonEKSLocalOutpostClusterPolicy e AmazonEKSLocalOutpostServiceRolePolicy
VPC e sottoreti del cluster	Consulta la sezione Requisiti e considerazioni su VPC e sottoreti di Amazon EKS	Consulta la sezione Requisiti e considerazioni su VPC e sottoreti del cluster locale Amazon EKS
Accesso all'endpoint del cluster	Pubblico o privato o entrambi	Solo privati
Autenticazione del server API Kubernetes	AWS Identity and Access Management (IAM) e OIDC	IAM e certificati `x.509`
Tipi di nodo	Solo autogestito	Solo autogestito
Tipi di elaborazione dei nodi	Amazon EC2 on demand	Amazon EC2 on demand

Funzionalità	Cluster esteso	Cluster locale
Tipi di archiviazione dei nodi	gp2 di Amazon EBS e SSD NVMe locali	gp2 di Amazon EBS e SSD NVMe locali
AMI ottimizzate per Amazon EKS	Amazon Linux, Windows e Bottlerocket	Solo Amazon Linux
Versioni IP	Solo `IPv4`	Solo `IPv4`
Componenti aggiuntivi	Componenti aggiuntivi Amazon EKS o autogestiti	Solo componenti aggiuntivi autogestiti
Interfaccia di rete del container predefinita	Amazon VPC CNI plugin for Kubernetes	Amazon VPC CNI plugin for Kubernetes
Registri del piano di controllo Kubernetes	Amazon CloudWatch Logs	Amazon CloudWatch Logs
Bilanciamento del carico	Utilizza il AWS Load Balancer Controller per allocare solo Application Load Balancer (nessun Network Load Balancer)	Utilizza il AWS Load Balancer Controller per allocare solo Application Load Balancer (nessun Network Load Balancer)
Crittografia a busta dei segreti	Consulta la sezione Abilitazione della crittografia segreta dei dati in transito su un cluster esistente	Non supportato
Ruoli IAM per gli account di servizio	Consulta la sezione Ruoli IAM per gli account di servizio	Non supportato
Risoluzione dei problemi	Consulta la sezione Risoluzione dei problemi di Amazon EKS	Consulta la sezione Risoluzione dei problemi relativi ai cluster locali per Amazon EKS su AWS Outposts

Cluster locali per Amazon EKS su AWS Outposts

I cluster locali consentono di eseguire l'intero cluster Amazon EKS in locale su AWS Outposts. Ciò consente di ridurre il rischio di tempi di inattività delle applicazioni che possono derivare da disconnessioni temporanee del cloud dalla rete. Queste disconnessioni di rete possono essere causate da interruzioni della fibra o da eventi meteorologici. Poiché l'intero cluster Kubernetes viene eseguito localmente su Outposts, le applicazioni rimangono disponibili. Durante le disconnessioni del cloud dalla rete è possibile eseguire operazioni del cluster. Per ulteriori informazioni, consulta Preparazione alle disconnessioni dalla rete. Il diagramma seguente mostra un'implementazione di un cluster locale.



I cluster locali sono generalmente disponibili per l'uso con i rack di Outposts.

Regioni AWS supportate

È possibile creare cluster locali nelle seguenti Regioni AWS: Stati Uniti orientali (Ohio), Stati Uniti orientali (Virginia settentrionale), Stati Uniti occidentali (California settentrionale), Stati Uniti occidentali (Oregon), Asia Pacifico (Seoul), Asia Pacifico (Singapore), Asia Pacifico (Sydney), Asia Pacifico (Tokyo), Canada (Centrale), Europa (Francoforte), Europa (Irlanda), Europa (Londra), Medio Oriente (Bahrein) e Sud America (San Paolo). Per informazioni dettagliate sulle funzionalità supportate, consulta la sezione Confronto tra le opzioni di implementazione.

Argomenti

Creazione di un cluster su un Outpost

Questo argomento offre una panoramica degli elementi da prendere in considerazione quando si esegue un cluster locale su un Outpost. L'argomento fornisce anche istruzioni su come implementare un cluster locale su un Outpost.

Considerazioni

Important

- Queste considerazioni non sono trattate nella documentazione correlata di Amazon EKS. Se altri argomenti della documentazione di Amazon EKS sono in conflitto con le considerazioni qui riportate, segui le considerazioni qui.
- Queste considerazioni sono soggette a modifiche e potrebbero cambiare frequentemente. Pertanto, ti consigliamo di rivedere regolarmente questo argomento.
- Molte delle considerazioni sono diverse da quelle relative alla creazione di un cluster sul Cloud AWS.

- I cluster locali supportano solo i rack dell'Outpost. Un singolo cluster locale può essere eseguito su più dell'Outpost fisici che comprendono un unico Outpost logico. Un singolo cluster locale non può essere eseguito su più Outpost logici. Ogni Outpost logico ha un singolo ARN dell'Outpost.
- I cluster locali eseguono e gestiscono il piano di controllo (control-plane) Kubernetes nel tuo account sull'Outpost. Non puoi eseguire carichi di lavoro sulle istanze del piano di controllo Kubernetes né modificare i componenti del piano di controllo Kubernetes. Questi nodi sono gestiti dal servizio Amazon EKS. Le modifiche al piano di controllo Kubernetes non persistono tra un'operazione di gestione automatica di Amazon EKS e l'altra, come l'applicazione di patch.
- I cluster locali supportano componenti aggiuntivi autogestiti e gruppi di nodi Amazon Linux autogestiti. I componenti aggiuntivi Amazon VPC CNI plugin for Kubernetes, `kube-proxy` e CoreDNS vengono installati automaticamente nei cluster locali.
- I cluster locali richiedono l'utilizzo di Amazon EBS su Outposts. Il tuo Outpost deve avere a disposizione Amazon EBS per l'archiviazione del piano di controllo Kubernetes.
- I cluster locali richiedono l'utilizzo di Amazon EBS su Outposts. Il tuo Outpost deve avere a disposizione Amazon EBS per l'archiviazione del piano di controllo Kubernetes. Gli Outpost supportano solo i volumi gp2 di Amazon EBS.

- I `PersistentVolumes` Kubernetes supportati da Amazon EBS sono supportati tramite il driver CSI per Amazon EBS.

Prerequisiti

- Familiarità con le opzioni di implementazione di Outposts, Considerazioni sulla capacità e Requisiti e considerazioni su VPC e sottoreti del cluster locale Amazon EKS.
- Un Outpost esistente. Per ulteriori informazioni, consulta Cos'è AWS Outposts.
- Lo strumento a riga di comando `kubectl` è installato sul computer o AWS CloudShell. La versione può essere uguale oppure immediatamente precedente o successiva alla versione Kubernetes del cluster. Ad esempio, se la versione del cluster è `1.29`, puoi usare `kubectl` versione `1.28`, `1.29` o `1.30`. Per installare o aggiornare `kubectl`, consulta Installazione o aggiornamento di `kubectl`:
- Versione `2.12.3` o successiva o versione `1.27.160` o successiva di AWS Command Line Interface (AWS CLI) installato e configurato sul dispositivo o. AWS CloudShell Per verificare la versione attuale, usa **`aws --version | cut -d / -f2 | cut -d ' ' -f1`**. I programmi di gestione dei pacchetti, come `yum`, `apt-get` o Homebrew per macOS, spesso sono aggiornati a versioni precedenti della AWS CLI. Per installare la versione più recente, consulta le sezioni Installazione, aggiornamento e disinstallazione della AWS CLI e Configurazione rapida con aws configure nella Guida per l'utente dell'AWS Command Line Interface . La AWS CLI versione installata in AWS CloudShell potrebbe anche contenere diverse versioni precedenti alla versione più recente. Per aggiornarla, consulta Installazione nella tua home directory nella Guida AWS CLI per l'AWS CloudShell utente.
- Un principale IAM (utente o ruolo) con autorizzazioni `create` e `describe` per un cluster Amazon EKS. Per ulteriori informazioni, consulta Creazione di un cluster Kubernetes locale su un Outpost e Elencare o descrivere tutti i cluster.

Quando viene creato un cluster Amazon EKS locale, il principale IAM che crea il cluster viene aggiunto in modo permanente. Il principale viene aggiunto specificamente alla tabella di autorizzazione RBAC Kubernetes come amministratore. Questa entità dispone di autorizzazioni `system:masters`. L'identità di questa entità non è visibile nella configurazione del cluster, per cui è fondamentale prendere nota dell'entità che ha creato il cluster, in modo da non eliminarla. Inizialmente, solo il principale che ha creato il server può effettuare chiamate al server API Kubernetes utilizzando `kubectl`. Se utilizzi la console per creare il cluster, assicurati che le stesse credenziali IAM siano presenti nella catena di credenziali AWS SDK quando `kubectl` esegui i

comandi sul cluster. Dopo aver creato il cluster, è possibile concedere l'accesso ad altri principali IAM al cluster.

Creazione di un cluster locale di Amazon EKS

È possibile creare un cluster locale con `eksctl`, la AWS Management Console, la AWS CLI, l'API Amazon EKS, gli SDK AWS, AWS CloudFormation o Terraform.

1. Crea un cluster locale.

   eksctl

   Prerequisito

   La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

   Per creare il tuo cluster con **`eksctl`**

   1. Copia i seguenti contenuti sul tuo dispositivo. Sostituisci i valori seguenti, quindi esegui il comando modificato per creare il file `outpost-control-plane.yaml`:
      - Sostituisci *`region-code`* con la Regione AWS supportata in cui desideri creare il cluster.
      - Sostituisci *`my-cluster`* con un nome da assegnare al cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può contenere più di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster. Il nome deve essere univoco all'interno Regione AWS e in Account AWS cui si sta creando il cluster.
      - Sostituisci *`vpc-ExampleID1`* e *`subnet-ExampleID1`* con gli ID del tuo VPC e della sottorete esistenti. Il VPC e la sottorete devono soddisfare i requisiti in Requisiti e considerazioni su VPC e sottoreti del cluster locale Amazon EKS.
      - *`uniqueid`*Sostituiscilo con l'ID del tuo Outpost.
      - Sostituisci *`m5.large`* con un tipo di istanza disponibile sul tuo Outpost. Prima di scegliere un tipo di istanza, consulta la sezione Considerazioni sulla capacità. Vengono implementate tre istanze del piano di controllo. Questo numero non può essere modificato.

```
cat >outpost-control-plane.yaml <<EOF
apiVersion: eksctl.io/v1alpha5
kind: ClusterConfig

metadata:
  name: my-cluster
  region: region-code
  version: "1.24"

vpc:
  clusterEndpoints:
    privateAccess: true
  id: "vpc-vpc-ExampleID1"
  subnets:
    private:
      outpost-subnet-1:
        id: "subnet-subnet-ExampleID1"

outpost:
  controlPlaneOutpostARN: arn:aws:outposts:region-code:111122223333:outpost/
op-uniqueid
  controlPlaneInstanceType: m5.large
EOF
```

Per un elenco completo di tutte le opzioni disponibili, consulta Supporto di AWS Outposts e Schema del file di configurazione nella documentazione di `eksctl`.

2. Crea il cluster utilizzando il file di configurazione creato nel passaggio precedente. `eksctl` crea un VPC e una sottorete sul tuo Outpost in cui implementare il cluster.

```
eksctl create cluster -f outpost-control-plane.yaml
```

Il provisioning del cluster richiede diversi minuti. Durante la creazione del cluster, vengono visualizzate diverse righe di output. L'ultima riga di output è simile alla seguente riga di esempio.

```
[#]  EKS cluster "my-cluster" in "region-code" region is ready
```

> **Tip**
>
> Per visualizzare la maggior parte delle opzioni che è possibile specificare durante la creazione di un cluster con `eksctl`, utilizza il comando **`eksctl create cluster --help`**. Per visualizzare tutte le opzioni disponibili, puoi utilizzare un file `config`. Per ulteriori informazioni, consulta Uso dei file config e lo Schema dei file config nella documentazione di `eksctl`. Gli esempi di file di configurazione sono disponibili su GitHub.

`Eksctl` ha creato automaticamente una voce di accesso per il principale IAM (utente o ruolo) che ha creato il cluster e ha concesso al principale IAM autorizzazioni di amministratore per gli oggetti Kubernetes sul cluster. Se non desideri che il creatore del cluster disponga di un accesso di amministratore agli oggetti Kubernetes sul cluster, aggiungi il seguente testo al file di configurazione precedente: **`bootstrapClusterCreatorAdminPermissions: false`** (allo stesso livello di `metadata`, `vpc` e `outpost`). Se hai aggiunto l'opzione, dopo la creazione del cluster è necessario creare una voce di accesso per almeno un principale IAM, altrimenti nessun principale IAM avrà accesso agli oggetti Kubernetes sul cluster.

AWS Management Console

Prerequisito

Un VPC e una sottorete esistenti che soddisfano i requisiti di Amazon EKS. Per ulteriori informazioni, consulta Requisiti e considerazioni su VPC e sottoreti del cluster locale Amazon EKS.

Per creare il tuo cluster con AWS Management Console

1. Se disponi già di un ruolo IAM del cluster locale o intendi creare il cluster con `eksctl`, puoi ignorare questo passaggio. Per impostazione predefinita, `eksctl` crea un ruolo per te.

   a. Per creare un file JSON della policy di attendibilità IAM, esegui il comando seguente.

```
cat >eks-local-cluster-role-trust-policy.json <<EOF
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
EOF
```

b. Crea il ruolo IAM del cluster Amazon EKS. Per creare un ruolo IAM, è necessario assegnare l'azione `iam:CreateRole` (autorizzazione) al principale IAM che sta creando il ruolo.

```
aws iam create-role --role-name myAmazonEKSLocalClusterRole --assume-role-
policy-document file://"eks-local-cluster-role-trust-policy.json"
```

c. Allega al ruolo la policy gestita da Amazon EKS, denominata `AmazonEKSLocalOutpostClusterPolicy`. Per allegare una policy IAM a un principale IAM, è necessario assegnare al principale che sta allegando la policy una delle azioni IAM (autorizzazioni) seguenti: `iam:AttachUserPolicy` o `iam:AttachRolePolicy`.

```
aws iam attach-role-policy --policy-arn arn:aws:iam::aws:policy/
AmazonEKSLocalOutpostClusterPolicy --role-name myAmazonEKSLocalClusterRole
```

2. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
3. Nella parte superiore dello schermo della console, assicurati di avere selezionato una Regione AWS supportata.
4. Seleziona Aggiungi cluster e quindi Crea.
5. Nella pagina Configure cluster (Configura cluster), inserisci o seleziona i valori per i campi seguenti:
   - Posizione del piano di controllo (control plane) Kubernetes: scegli AWS Outposts.
   - Outpost ID (ID Outpost): scegli l'ID dell'Outpost su cui vuoi creare il tuo piano di controllo.
   - Instance type (Tipo di istanza): seleziona un tipo di istanza. Vengono visualizzati solo i tipi di istanza disponibili nel tuo Outpost. Nell'elenco a discesa, ogni tipo di istanza

descrive per quanti nodi è consigliato il tipo di istanza. Prima di scegliere un tipo di istanza, consulta la sezione Considerazioni sulla capacità. Tutte le repliche vengono implementate utilizzando lo stesso tipo di istanza. Non è possibile modificare il tipo di istanza dopo la creazione del cluster. Vengono implementate tre istanze del piano di controllo. Questo numero non può essere modificato.

- Nome: un nome per il cluster. Deve essere unico nel tuo Account AWS. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster. Il nome deve essere univoco all'interno Regione AWS e in Account AWS cui si sta creando il cluster.
- Versione Kubernetes: scegli la versione di Kubernetes da utilizzare per il cluster. È preferibile selezionare la versione più recente, a meno che non occorra utilizzare una versione precedente.
- Ruolo del servizio cluster: scegli il ruolo IAM del cluster Amazon EKS creato in un passaggio precedente per consentire al piano di Kubernetes controllo di gestire AWS le risorse.
- Accesso di amministratore del cluster Kubernetes: se desideri che il principale IAM (ruolo o utente) che crea il cluster disponga dell'accesso di amministratore agli oggetti Kubernetes sul cluster, accetta l'impostazione predefinita (consenti). Amazon EKS crea una voce di accesso per il principale IAM e concede le autorizzazioni di amministratore del cluster per la voce di accesso. Per ulteriori informazioni sulle voci di accesso, consulta la sezione Gestire le voci di accesso.

  Se desideri che un principale IAM diverso da quello che crea il cluster disponga dell'accesso di amministratore agli oggetti Kubernetes del cluster, scegli l'opzione "Non consentire". Dopo la creazione del cluster, qualsiasi principale IAM che disponga delle autorizzazioni IAM per creare voci di accesso può aggiungere una voce di accesso per tutti i principali IAM che necessitano dell'accesso agli oggetti Kubernetes del cluster. Per ulteriori informazioni sulle autorizzazioni IAM richieste, consulta Actions defined by Amazon Elastic Kubernetes Service nella Documentazione di riferimento per l'autorizzazione ai servizi. Se scegli l'opzione "non consentire" e non crei alcuna voce di accesso, nessun principale IAM avrà accesso agli oggetti Kubernetes sul cluster.
- Tag (Facoltativo): aggiunge eventuali tag al cluster. Per ulteriori informazioni, consulta Assegnazione di tag alle risorse Amazon EKS.

Al termine, seleziona Avanti.

6. Nella pagina Specifica reti, seleziona i valori dei campi riportati di seguito:
   - VPC: scegli un VPC esistente. Il VPC deve disporre di un numero sufficiente di indirizzi IP da mettere a disposizione per il cluster, i nodi e le altre risorse Kubernetes da creare. Il VPC deve soddisfare i requisiti elencati in Considerazioni e requisiti relativi al VPC.
   - Sottoreti: per impostazione predefinita, tutte le sottoreti disponibili nel VPC specificato nel campo precedente sono preselezionate. Le sottoreti scelte devono soddisfare i requisiti elencati in Considerazioni e requisiti relativi alle sottoreti.

     Gruppi di sicurezza: (facoltativo) specifica uno o più gruppi di sicurezza da associare alle interfacce di rete create da Amazon EKS. Amazon EKS crea automaticamente un gruppo di sicurezza che consente la comunicazione tra il cluster e il VPC. Amazon EKS associa questo gruppo di sicurezza, e altri gruppi eventualmente scelti dall'utente, alle interfacce di rete create. Per ulteriori informazioni sul gruppo di sicurezza del cluster creato da Amazon EKS, consulta Considerazioni e requisiti relativi al gruppo di sicurezza Amazon EKS. Puoi modificare le regole nel gruppo di sicurezza del cluster creato da Amazon EKS. Se vuoi aggiungere i tuoi gruppi di sicurezza, non potrai modificare i gruppi scelti dopo la creazione del cluster. Per consentire agli host on-premise di comunicare con l'endpoint del cluster, è necessario consentire il traffico in entrata dal gruppo di sicurezza del cluster. Per i cluster che non dispongono di una connessione Internet in ingresso e in uscita (noti anche come cluster privati), è necessario effettuare una delle seguenti operazioni:
     - Aggiungi il gruppo di sicurezza associato agli endpoint VPC richiesti. Per ulteriori informazioni sugli endpoint richiesti, consulta Endpoint VPC di interfaccia in Accesso della sottorete ai Servizi AWS.
     - Modifica il gruppo di sicurezza creato da Amazon EKS per consentire il traffico proveniente dal gruppo di sicurezza associato agli endpoint VPC.

   Quando hai finito con la pagina, seleziona Avanti.
7. Nella pagina Configura osservabilità, puoi scegliere facoltativamente le opzioni Parametri e Registrazione del piano di controllo che desideri attivare. Per impostazione predefinita, i tipi di log sono disattivati.
   - Per ulteriori informazioni sull'opzione relativa ai parametri Prometheus, consulta Attivazione dei parametri Prometheus durante la creazione di un cluster.

- Per ulteriori informazioni sulle opzioni Registrazione del piano di controllo, consulta Logging del piano di controllo di Amazon EKS.

Al termine, seleziona Avanti.

8. Nella pagina Rivedi e crea, controlla le informazioni che hai inserito o selezionato nelle pagine precedenti. Se devi apportare modifiche, seleziona Edit (Modifica). Al termine della configurazione, seleziona Create (Crea). Durante il provisioning del cluster, nel campo Stato viene visualizzato il messaggio CREAZIONE.

   Il provisioning del cluster richiede diversi minuti.

2. Dopo avere creato il cluster, puoi visualizzare le istanze del piano di controllo Amazon EC2 che sono state create.

```
aws ec2 describe-instances --query 'Reservations[*].Instances[*].{Name:Tags[?
Key==`Name`]|[0].Value}' | grep my-cluster-control-plane
```

Di seguito viene riportato un output di esempio:

```
"Name": "my-cluster-control-plane-id1"
"Name": "my-cluster-control-plane-id2"
"Name": "my-cluster-control-plane-id3"
```

Ogni istanza è contaminata dal taint `node-role.eks-local.amazonaws.com/control-plane` per evitare che i carichi di lavoro vengano pianificati sulle istanze del piano di controllo. Per ulteriori informazioni sui taint, consulta la sezione Taints and Tolerations (Taint e tolleranze) nella documentazione di Kubernetes. Amazon EKS monitora continuamente lo stato dei cluster locali. Eseguiamo operazioni di gestione automatiche, come applicazioni di patch di sicurezza e riparazioni di istanze non integre. Quando i cluster locali vengono disconnessi dal cloud, eseguiamo una serie di operazioni per garantire che il cluster venga ripristinato a uno stato di integrità al momento della riconnessione.

3. Se hai creato il cluster utilizzando `eksctl`, puoi ignorare questo passaggio. `eksctl` lo completerà automaticamente per tuo conto. Abilita `kubectl` per consentire la comunicazione con il cluster aggiungendo un nuovo contesto al file `config` `kubectl`. Per istruzioni su come creare e aggiornare il file, consulta Creazione o aggiornamento di un file `kubeconfig` per un cluster Amazon EKS.

```
aws eks update-kubeconfig --region region-code --name my-cluster
```

Di seguito viene riportato un output di esempio:

```
Added new context arn:aws:eks:region-code:111122223333:cluster/my-cluster to /home/
username/.kube/config
```

4. Per collegarti al server API Kubernetes del tuo cluster locale, devi disporre dell'accesso al gateway locale per la sottorete o collegarti dall'interno del VPC. Per ulteriori informazioni sul collegamento di un rack Outpost alla rete locale, consulta How local gateway for rack work nella Guida per l' AWS Outposts utente. Se utilizzi l'instradamento VPC diretto e la sottorete dell'Outpost ha un instradamento al gateway locale, gli indirizzi IP privati delle istanze del piano di controllo Kubernetes vengono trasmessi automaticamente sulla rete locale. L'endpoint del server API Kubernetes del cluster locale è ospitato in Amazon Route 53 (Route 53). L'endpoint del servizio API può essere risolto dai server DNS pubblici negli indirizzi IP privati dei server dell'API Kubernetes.

   Le istanze del piano di controllo (control plane) Kubernetes dei cluster locali sono configurate con interfacce di rete Elastic statiche con indirizzi IP privati fissi che non cambiano durante il ciclo di vita del cluster. Le macchine che interagiscono con il server API Kubernetes potrebbero non avere connettività a Route 53 durante le disconnessioni di rete. In tal caso, si consiglia di configurare `/etc/hosts` con gli indirizzi IP privati statici per continuare le operazioni. Ti consigliamo inoltre di configurare i server DNS locali e di collegarli al tuo Outpost. Per ulteriori informazioni, consulta la documentazione relativa ad AWS Outposts. Esegui il comando riportato per confermare che è stata stabilita la comunicazione con il cluster.

   ```
   kubectl get svc
   ```

   Di seguito viene riportato un output di esempio:

   ```
   NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
   kubernetes   ClusterIP   10.100.0.1   <none>        443/TCP   28h
   ```

5. (Facoltativo) Verifica l'autenticazione al cluster locale quando è disconnesso dal Cloud AWS. Per istruzioni, consulta Preparazione alle disconnessioni dalla rete.

Risorse interne

Amazon EKS crea le seguenti risorse sul tuo cluster. Le risorse sono destinate all'uso interno di Amazon EKS. Per garantire il corretto funzionamento del cluster, non modificare o alterare queste risorse.

- I seguenti Pods mirror:
  - `aws-iam-authenticator-`*`node-hostname`*
  - `eks-certificates-controller-`*`node-hostname`*
  - `etcd-`*`node-hostname`*
  - `kube-apiserver-`*`node-hostname`*
  - `kube-controller-manager-`*`node-hostname`*
  - `kube-scheduler-`*`node-hostname`*
- I seguenti componenti aggiuntivi autogestiti:
  - `kube-system/coredns`
  - `kube-system/kube-proxy` (non viene creato finché non si aggiunge il primo nodo)
  - `kube-system/aws-node` (non viene creato finché non si aggiunge il primo nodo). I cluster locali utilizzano il plugin Amazon VPC CNI plugin for Kubernetes per le reti di cluster. Non modificare la configurazione per le istanze del piano di controllo (pod denominati `aws-node-controlplane-*`). Ci sono variabili di configurazione che consentono di modificare il valore predefinito quando il plugin crea nuove interfacce di rete. Per ulteriori informazioni, consulta la documentazione su. GitHub
- I seguenti servizi:
  - `default/kubernetes`
  - `kube-system/kube-dns`
- Una `PodSecurityPolicy` denominata `eks.system`
- Una `ClusterRole` denominata `eks:system:podsecuritypolicy`
- Una `ClusterRoleBinding` denominata `eks:system`
- Una PodSecuritypolitica predefinita
- Oltre al gruppo di sicurezza del cluster, Amazon EKS crea un gruppo di sicurezza denominato all'interno del tuo Account AWS account`eks-local-internal-do-not-use-or-edit-`*`cluster-name`*`-`*`uniqueid`*. Questo gruppo di sicurezza consente al traffico di fluire liberamente tra i componenti Kubernetes in esecuzione sulle istanze del piano di controllo.

Fasi successive consigliate:

- Concedi al responsabile IAM che ha creato il cluster le autorizzazioni necessarie per visualizzare Kubernetes le risorse in AWS Management Console
- Concedere alle entità IAM l'accesso al cluster. Per consentire alle entità di visualizzare le risorse Kubernetes nella console Amazon EKS, concedi le relative Autorizzazioni richieste.
- Configurare la registrazione per il cluster
- Acquisisci familiarità con ciò che accade durante le disconnessioni dalla rete.
- Aggiungere nodi al cluster
- Prendi in considerazione la possibilità di impostare un piano di backup per `etcd`. Amazon EKS non supporta il backup e il ripristino automatici di `etcd` per i cluster locali. Per ulteriori informazioni, consulta la sezione Backing up an `etcd` cluster (Eseguire il backup di un cluster) nella documentazione di Kubernetes. Le due opzioni principali sono l'utilizzo di `etcdctl` per automatizzare l'acquisizione di snapshot o l'utilizzo del backup del volume di archiviazione di Amazon EBS.

Versioni della piattaforma del cluster locale Amazon EKS

Le versioni della piattaforma del cluster locale rappresentano le funzionalità del cluster Amazon EKS su AWS Outposts. Le versioni includono i componenti che vengono eseguiti sul piano di controllo Kubernetes, i cui flag del server API Kubernetes sono abilitati. Includono anche la versione corrente della patch Kubernetes. Ogni versione secondaria di Kubernetes dispone di una o più versioni della piattaforma associate. Le versioni della piattaforma per versioni secondarie di Kubernetes diverse sono indipendenti. Le versioni della piattaforma per i cluster locali e i cluster Amazon EKS nel cloud sono indipendenti.

Quando è disponibile una nuova versione secondaria di Kubernetes per i cluster locali, ad esempio `1.28`, la versione della piattaforma iniziale per tale versione secondaria di Kubernetes inizia da `eks-local-outposts.1`. Tuttavia, Amazon EKS rilascia periodicamente nuove versioni della piattaforma per abilitare nuove impostazioni del piano di controllo Kubernetes e fornire soluzioni per problemi relativi alla sicurezza.

Quando nuove versioni della piattaforma del cluster locale diventano disponibili per una versione secondaria:

- Il numero di versione della piattaforma viene incrementato (`eks-local-outposts.n+1`).

- Amazon EKS aggiorna automaticamente tutti i cluster locali esistenti alla versione della piattaforma più recente per la versione secondaria Kubernetes corrispondente. Gli aggiornamenti automatici delle versioni della piattaforma esistenti vengono implementati in modo incrementale. Il processo di implementazione potrebbe richiedere alcuni minuti. Se le ultime caratteristiche della versione della piattaforma sono richieste immediatamente, consigliamo di creare un nuovo cluster locale.
- Amazon EKS potrebbe pubblicare un nuovo nodo AMI con una versione patch corrispondente. Tutte le versioni della patch sono compatibili tra il piano di controllo Kubernetes e le AMI dei nodi per una data versione secondaria di Kubernetes.

Nuove versioni della piattaforma non introducono modifiche in conflitto né causano interruzioni del servizio.

I cluster vengono sempre creati con la versione della piattaforma disponibile più recente (`eks-local-outposts.n`) per la versione Kubernetes specificata.

Le versioni della piattaforma correnti e recenti sono descritte nelle tabelle qui di seguito.

Kubernetes versione **`1.28`**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.28`: `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `DefaultStorageClass`, `DefaultTolerationSeconds`, `ExtendedResourceToleration`, `LimitRanger`, `MutatingAdmissionWebhook`, `NamespaceLifecycle`, `NodeRestriction`, `PersistentVolumeClaimResize`, `Priority`, `PodSecurity`, `ResourceQuota`, `RuntimeClass`, `ServiceAccount`, `StorageObjectInUseProtection`, `TaintNodesByCondition`, `ValidatingAdmissionPolicy`, e `ValidatingAdmissionWebhook`.

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
`1.28.6`	`eks-local-outposts.5`	Versione Bottlerocket aggiornata alla v1.19.3 contenente le correzioni di bug più recenti per supportare l'avvio locale in Outposts.	18 aprile 2024

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.28.6	eks-local-outposts.4	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti. Supporto ripristinato o avvio locale in Outposts. BottlerocketVersione ridotta a per compatibilità. v1.15.1	2 aprile 2024
1.28.6	eks-local-outposts.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	22 marzo 2024
1.28.6	eks-local-outposts.2	Nuova versione della piattaforma con correzioni e miglioramenti di sicurezza kube-proxy aggiornata a. v1.28.6 AWS IAM v0.6.17 Authenticator aggiornato a. Plugin Amazon VPC CNI per Kubernetes sottoposto a downgrade per motivi di compatibilità. v1.13.2 Versione aggiornata a. Bottlerocket v1.19.2	8 marzo 2024
1.28.1	eks-local-outposts.1	Versione iniziale della versione Kubernetes per cluster v1.28 Amazon EKS locali su Outpost	4 ottobre 2023

Kubernetes versione **1.27**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma 1.27: CertificateApproval, CertificateSigning, CertificateSubjectRestriction, DefaultIngressClass, DefaultStorageClass, DefaultTolerationSeconds, ExtendedResourceToleration, LimitRanger, MutatingAdmissionWebhook,

NamespaceLifecycle, NodeRestriction, PersistentVolumeClaimResize, Priority, PodSecurity, ResourceQuota, RuntimeClass, ServiceAccount, StorageObjectInUseProtection, TaintNodesByCondition, ValidatingAdmissionPolicy, e ValidatingAdmissionWebhook.

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.27.10	eks-local-outposts.5	Nuova piattaforma con correzioni e miglioramenti di sicurezza.	2 aprile 2024
1.27.10	eks-local-outposts.4	Nuova piattaforma con correzioni e miglioramenti di sicurezza. kube-proxy aggiornato a. v1.27.10 AWS IAM v0.6.17 Authenticator aggiornato a. BottlerocketVersione aggiornata a. v1.19.2	22 marzo 2024
1.27.3	eks-local-outposts.3	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti. kube-proxy aggiornata a v1.27.3. Plug-in CNI di Amazon VPC per Kubernetes aggiornato a v1.13.2.	14 luglio 2023
1.27.1	eks-local-outposts.2	Immagine CoreDNS aggiornata a v1.10.1	22 giugno 2023
1.27.1	eks-local-outposts.1	Versione iniziale della versione Kubernetes per cluster Amazon EKS 1.27 locali su Outposts.	30 maggio 2023

Kubernetes versione **1.26**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.26`: `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `DefaultStorageClass`, `DefaultTolerationSeconds`, `ExtendedResourceToleration`, `LimitRanger`, `MutatingAdmissionWebhook`, `NamespaceLifecycle`, `NodeRestriction`, `PersistentVolumeClaimResize`, `Priority`, `PodSecurity`, `ResourceQuota`, `RuntimeClass`, `ServiceAccount`, `StorageObjectInUseProtection`, `TaintNodesByCondition`, `ValidatingAdmissionPolicy`, e `ValidatingAdmissionWebhook`.

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
`1.26.13`	`eks-local-outposts.5`	Nuova versione della piattaforma con correzioni e miglioramenti alla sicurezza. kube-proxy aggiornato a. `v1.26.13` AWS IAM `v0.6.17` Authenticator aggiornato a. BottlerocketVersione aggiornata a. `v1.19.2`	22 marzo 2024

Kubernetes versione **1.25**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma `1.25`: `CertificateApproval`, `CertificateSigning`, `CertificateSubjectRestriction`, `DefaultIngressClass`, `DefaultStorageClass`, `DefaultTolerationSeconds`, `ExtendedResourceToleration`, `LimitRanger`, `MutatingAdmissionWebhook`, `NamespaceLifecycle`, `NodeRestriction`, `PersistentVolumeClaimResize`, `Priority`, `PodSecurity`, `ResourceQuota`, `RuntimeClass`, `ServiceAccount`, `StorageObjectInUseProtection`, `TaintNodesByCondition` e `ValidatingAdmissionWebhook`.

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.25.16	eks-local-outposts.7	Nuova versione della piattaforma con correzioni e miglioramenti di sicurezza. kube-proxy aggiornato a. v1.25.16 AWS IAM v0.6.17 Authenticator aggiornato a. BottlerocketVersione aggiornata a. v1.19.2	22 marzo 2024
1.25.11	eks-local-outposts.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti. kube-proxy aggiornata a v1.25.11. Plug-in CNI di Amazon VPC per Kubernetes aggiornato a v1.13.2.	14 luglio 2023
1.25.9	eks-local-outposts.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	13 luglio 2023
1.25.6	eks-local-outposts.4	Versione Bottlerocket aggiornata a 1.13.2	2 maggio 2023
1.25.6	eks-local-outposts.3	Sistema operativo dell'istanza del piano di controllo Amazon EKS aggiornato alla versione Bottlerocket e plug-in v1.13.1 Amazon VPC CNI per Kubernetes aggiornato alla versione. v1.12.6	14 aprile 2023

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.25.6	eks-local-outposts.2	Raccolta di dati diagnostici migliorata per le istanze del piano di controllo Kubernetes.	8 marzo 2023
1.25.6	eks-local-outposts.1	Versione iniziale della versione Kubernetes per cluster Amazon EKS 1.25 locali su Outposts.	1 marzo 2023

Kubernetes versione **1.24**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma 1.24: DefaultStorageClass, DefaultTolerationSeconds, LimitRanger, MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, ResourceQuota, ServiceAccount, ValidatingAdmissionWebhook, PodSecurityPolicy, TaintNodesByCondition, StorageObjectInUseProtection, PersistentVolumeClaimResize, ExtendedResourceToleration, CertificateApproval, PodPriority, CertificateSigning, CertificateSubjectRestriction, RuntimeClass e DefaultIngressClass.

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.24.17	eks-local-outposts.7	Nuova versione della piattaforma con correzioni e miglioramenti alla sicurezza. kube-proxy aggiornato a. v1.25.16 AWS v0.6.17IAM Authenticator aggiornato. BottlerocketVersione aggiornata a. v1.19.2	22 marzo 2024
1.24.15	eks-local-outposts.6	Nuova versione della piattaforma con correzioni di sicurezza	14 luglio 2023

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
		e miglioramenti. kube-proxy aggiornata a v1.24.15. Plug-in CNI di Amazon VPC per Kubernetes aggiornato a v1.13.2.	
1.24.13	eks-local-outposts.5	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	13 luglio 2023
1.24.9	eks-local-outposts.4	Versione Bottlerocket aggiornata a 1.13.2	2 maggio 2023
1.24.9	eks-local-outposts.3	Sistema operativo dell'istanza del piano di controllo Amazon EKS aggiornato alla versione Bottlerocket e plug-in v1.13.1 Amazon VPC CNI per Kubernetes aggiornato alla versione. v1.12.6	14 aprile 2023
1.24.9	eks-local-outposts.2	Raccolta di dati diagnostici migliorata per le istanze del piano di controllo Kubernetes.	8 marzo 2023
1.24.9	eks-local-outposts.1	Versione iniziale della versione Kubernetes per cluster Amazon EKS 1.24 locali su Outposts.	17 gennaio 2023

Kubernetes versione **1.23**

I seguenti controller di ammissione sono abilitati per tutte le versioni della piattaforma 1.23: DefaultStorageClass, DefaultTolerationSeconds, LimitRanger,

MutatingAdmissionWebhook, NamespaceLifecycle, NodeRestriction, ResourceQuota, ServiceAccount, ValidatingAdmissionWebhook, PodSecurityPolicy, TaintNodesByCondition, StorageObjectInUseProtection, PersistentVolumeClaimResize, ExtendedResourceToleration, CertificateApproval, PodPriority, CertificateSigning, CertificateSubjectRestriction, RuntimeClass e DefaultIngressClass.

Versione Kubernetes	Versione della piattaforma Amazon EKS	Note di rilascio	Data di rilascio
1.23.17	eks-local-outposts.6	Nuova versione della piattaforma con correzioni di sicurezza e miglioramenti.	13 luglio 2023
1.23.17	eks-local-outposts.5	Nuova versione della piattaforma con correzioni e miglioramenti alla sicurezza. kube-proxy aggiornato a. v1.23.17 BottlerocketVersione v1.14.1 aggiornata a.	6 luglio 2023
1.23.15	eks-local-outposts.4	Sistema operativo dell'istanza del piano di controllo Amazon EKS aggiornato alla versione Bottlerocket e plug-in v1.13.1 Amazon VPC CNI per Kubernetes aggiornato alla versione. v1.12.6	14 aprile 2023
1.23.15	eks-local-outposts.3	Raccolta di dati diagnostici migliorata per le istanze del piano di controllo Kubernetes.	8 marzo 2023
1.23.15	eks-local-outposts.2	Versione iniziale della versione Kubernetes per cluster Amazon EKS 1.23 locali su Outposts.	17 gennaio 2023

Requisiti e considerazioni su VPC e sottoreti del cluster locale Amazon EKS

Durante la creazione di un cluster locale, in genere si specificano un VPC e almeno una sottorete privata in esecuzione su Outposts. Questo argomento offre una panoramica dei requisiti e delle considerazioni relativi al VPC e alle sottoreti utilizzate con il cluster locale.

Considerazioni e requisiti relativi al VPC

Durante la creazione di un cluster locale, il VPC specificato deve soddisfare i requisiti e le considerazioni seguenti:

- Assicurati che il VPC disponga di un numero sufficiente di indirizzi IP per il cluster locale, i nodi e altre risorse Kubernetes che desideri creare. Se il VPC che desideri utilizzare non ha un numero sufficiente di indirizzi IP, aumenta tale numero. Tale operazione può essere effettuata associando i blocchi di instradamento interdominio senza classi (CIDR) secondari al VPC. L'associazione di blocchi di CIDR privati (RFC 1918) e pubblici (non RFC 1918) può avvenire prima o dopo la creazione del cluster. Il riconoscimento del blocco CIDR associato a un VPC da parte del cluster può richiedere fino a cinque ore.
- Il VPC non può avere prefissi IP o blocchi CIDR IPv6 assegnati. A causa di queste limitazioni, le informazioni che sono riportate in Aumentare la quantità di indirizzi IP disponibili per i nodi Amazon EC2 e `IPv6`indirizzi per cluster Pods e services non sono applicabili al tuo VPC.
- Il VPC ha un nome host DNS e una risoluzione DNS abilitati. Senza queste funzionalità la creazione del cluster locale ha esito negativo, quindi dovrai abilitare le funzionalità e creare nuovamente il cluster. Per ulteriori informazioni, consulta Attributi DNS nel VPC nella Guida per l'utente di Amazon VPC.
- Per accedere al cluster locale tramite la rete locale, il VPC deve essere associato alla tabella di instradamento del gateway locale dell'Outpost. Per ulteriori informazioni, consulta la sezione VPC associations (Associazioni del VPC) nella Guida per l'utente di AWS Outposts.

Considerazioni e requisiti relativi alle sottoreti

Durante la creazione del cluster, è necessario specificare almeno una sottorete privata. Se si specificano più sottoreti, le istanze del piano di controllo (control-plane) Kubernetes sono distribuite uniformemente tra le sottoreti. Se viene specificata più di una sottorete, le sottoreti devono essere presenti sullo stesso Outpost. Tuttavia, per comunicare tra loro le sottoreti devono avere gli instradamenti corretti e le autorizzazioni del gruppo di sicurezza. Quando crei un cluster locale, le sottoreti specificate devono soddisfare i requisiti seguenti:

- Le sottoreti si trovano tutte sullo stesso Outpost logico.
- Le sottoreti devono avere complessivamente almeno tre indirizzi IP disponibili per le istanze del piano di controllo Kubernetes. Se sono specificate tre sottoreti, ogni sottorete deve avere almeno un indirizzo IP disponibile. Se sono specificate due sottoreti, ogni sottorete deve avere almeno due indirizzi IP disponibili. Se è specificata una sottorete, la sottorete deve avere almeno tre indirizzi IP disponibili.
- Le sottoreti hanno un instradamento al gateway locale del rack Outpost per accedere al server API Kubernetes sulla rete locale. Se le sottoreti non hanno un instradamento per il gateway locale del rack Outpost, devi comunicare con il server API Kubernetes dall'interno del VPC.
- Le sottoreti devono utilizzare la denominazione basata sull'indirizzo IP. La denominazione basata sulle risorse di Amazon EC2 non è supportata da Amazon EKS.

Accesso della sottorete ai Servizi AWS

Le sottoreti private del cluster locale su Outposts devono essere in grado di comunicare con i Servizi AWS regionali. Ciò è possibile utilizzando un gateway NAT per l'accesso a Internet in uscita o, se si desidera mantenere privato tutto il traffico all'interno del VPC, tramite gli endpoint VPC di interfaccia.

Utilizzo di un gateway NAT

Le sottoreti private del cluster locale su Outposts devono avere una tabella di instradamento associata con una route a un gateway NAT in una sottorete pubblica che si trova nella zona di disponibilità principale dell'Outpost. La sottorete pubblica deve disporre di una route a un gateway Internet. Il gateway NAT consente l'accesso a Internet in uscita e impedisce le connessioni in entrata non richieste da Internet alle istanze sull'Outpost.

Utilizzo di endpoint VPC dell'interfaccia

Se le sottoreti private del cluster locale su Outposts non dispongono di una connessione Internet in uscita o se si desidera mantenere tutto il traffico privato all'interno del VPC, è necessario creare i seguenti endpoint VPC ed endpoint gateway in una sottorete regionale prima di creare il cluster.

Endpoint	Tipo di endpoint
`com.amazonaws.` *`region-code`* `.ssm`	Interfaccia
`com.amazonaws.` *`region-code`* `.ssmmessages`	Interfaccia

Endpoint	Tipo di endpoint
com.amazonaws. *region-code* .ec2messages	Interfaccia
com.amazonaws. *region-code* .ec2	Interfaccia
com.amazonaws. *region-code* .secretsmanager	Interfaccia
com.amazonaws. *region-code* .logs	Interfaccia
com.amazonaws. *region-code* .sts	Interfaccia
com.amazonaws. *region-code* .ecr.api	Interfaccia
com.amazonaws. *region-code* .ecr.dkr	Interfaccia
com.amazonaws. *region-code* .s3	Gateway

Gli endpoint devono soddisfare i seguenti requisiti:

- Creato in una sottorete privata situata nella zona di disponibilità principale dell'Outpost
- Abilitazione dei nomi DNS privati
- Disponi di un gruppo di sicurezza collegato che consente il traffico HTTPS in ingresso dall'intervallo CIDR della sottorete privata dell'Outpost.

La creazione di endpoint comporta dei costi. Per ulteriori informazioni, consulta Prezzi di AWS PrivateLink. Se i Pods hanno bisogno di accedere ad altri Servizi AWS, dovrai creare degli endpoint aggiuntivi. Per un elenco completo degli endpoint, consulta Servizi AWS che si integrano con AWS PrivateLink.

Crea un VPC

Puoi creare un VPC che soddisfi i requisiti precedenti utilizzando uno dei seguenti modelli AWS CloudFormation:

- Modello 1: questo modello crea un VPC con una sottorete privata nell'Outpost e una sottorete pubblica in Regione AWS. La sottorete privata ha un instradamento verso Internet tramite un

gateway NAT che si trova nella sottorete pubblica in Regione AWS. Questo modello può essere utilizzato per creare un cluster locale in una sottorete con accesso a Internet in uscita.

- Modello 2: questo modello crea un VPC con una sottorete privata sull'Outpost e il set minimo di endpoint VPC necessari per creare un cluster locale in una sottorete che non dispone di accesso a Internet in ingresso o in uscita (nota anche come sottorete privata).

Preparazione alle disconnessioni dalla rete

Se la tua rete locale ha perso la connettività con il Cloud AWS, puoi continuare a utilizzare il cluster Amazon EKS locale su un Outpost. Questo argomento illustra come preparare il cluster locale per le disconnessioni di rete e le considerazioni correlate.

Considerazioni sulla preparazione del cluster locale per una disconnessione di rete:

- I cluster locali consentono la stabilità e la continuità operatività durante le disconnessioni temporanee e non pianificate dalla rete. AWS Outposts rimane un'offerta completamente connessa che funge da estensione del Cloud AWS nel tuo data center. In caso di disconnessioni dalla rete tra l'Outpost e il Cloud AWS, ti consigliamo di provare a ripristinare la connessione. Per le relative istruzioni, consulta la lista di controllo per la risoluzione dei problemi di rete di un rack AWS Outposts nella Guida per l'utente di AWS Outposts. Per informazioni su come risolvere i problemi con i cluster locali, consulta Risoluzione dei problemi relativi ai cluster locali per Amazon EKS su AWS Outposts.
- Gli Outpost emettono un parametro `ConnectedStatus` che puoi utilizzare per monitorare lo stato di connettività del tuo Outpost. Per ulteriori informazioni, consulta la sezione Parametri dell'Outpost nella Guida per l'utente di AWS Outposts.
- I cluster locali utilizzano IAM come meccanismo di autenticazione predefinito tramite AWS Identity and Access Management Authenticator per Kubernetes. IAM non è disponibile durante le disconnessioni dalla rete. Pertanto, i cluster locali supportano un meccanismo di autenticazione alternativo basato sui certificati `x.509` che puoi utilizzare per connetterti al cluster durante le disconnessioni dalla rete. Per informazioni su come ottenere e utilizzare un certificato `x.509` per il tuo cluster, consulta Autenticazione al cluster locale durante una disconnessione dalla rete.
- Se non riesci ad accedere a Route 53 durante le disconnessioni dalla rete, prendi in considerazione l'utilizzo dei server DNS locali nel tuo ambiente on-premise. Le istanze del piano di controllo (control-plane) Kubernetes utilizzano indirizzi IP statici. Puoi configurare gli host che utilizzi per connetterti al cluster con il nome host e gli indirizzi IP dell'endpoint anziché utilizzare i server DNS locali. Per ulteriori informazioni, consulta DNS nella Guida per l'utente AWS Outposts.

- Se prevedi un aumento del traffico delle applicazioni durante le disconnessioni dalla rete, puoi allocare capacità di elaborazione di riserva nel tuo cluster quando sei connesso al cloud. Le istanze Amazon EC2 sono incluse nel prezzo di AWS Outposts. Pertanto, l'esecuzione di istanze di riserva non influisce sui costi di utilizzo di AWS.
- Durante le disconnessioni dalla rete, per consentire le operazioni di creazione, aggiornamento e dimensionamento dei carichi di lavoro, le immagini dei container dell'applicazione devono essere accessibili sulla rete locale e la capacità del cluster deve essere sufficiente. I cluster locali non ospitano un registro dei container per tuo conto. Se in precedenza i Pods sono stati eseguiti su tali nodi, le immagini di container vengono memorizzate nella cache dei nodi. Se generalmente estrai le immagini dei container dell'applicazione da Amazon ECR nel cloud, valuta l'esecuzione di una cache o un registro locale. Una cache o un registro locale è utile se hai bisogno di creare, aggiornare e dimensionare le risorse del carico di lavoro durante le disconnessioni dalla rete.
- I cluster locali utilizzano Amazon EBS come classe di archiviazione predefinita per i volumi persistenti e il driver Amazon EBS CSI per gestire il ciclo di vita dei volumi persistenti di Amazon EBS. Durante le disconnessioni dalla rete, i Pods supportati da Amazon EBS non possono essere creati, aggiornati o dimensionati. Questo perché queste operazioni richiedono chiamate all'API Amazon EBS nel cloud. Se stai implementando carichi di lavoro stateful su cluster locali e hai bisogno di creare, aggiornare o dimensionare le operazioni durante le disconnessioni dalla rete, prendi in considerazione l'utilizzo di un meccanismo di archiviazione alternativo.
- Gli snapshot di Amazon EBS non possono essere creati o eliminati se AWS Outposts non può accedere alle API pertinenti nella regione AWS (ad esempio le API per Amazon EBS o Amazon S3).
- Quando integri ALB (ingresso) con AWS Certificate Manager (ACM), i certificati vengono inviati e archiviati nella memoria dell'istanza AWS Outposts ALB Compute. Il terminale TLS corrente continuerà a funzionare in caso di disconnessione dalla Regione AWS. La modifica delle operazioni in questo contesto non riuscirà (ad esempio nuove definizioni di ingresso, nuove operazioni dell'API su certificati ACM, dimensionamento del calcolo ALB o rotazione dei certificati). Per ulteriori informazioni, consulta Soluzione dei problemi del rinnovo gestito dei certificati nella Guida per l'utente AWS Certificate Manager.
- I registri del piano di controllo Amazon EKS vengono memorizzati nella cache locale sulle istanze del piano di controllo Kubernetes durante le disconnessioni dalla rete. Al momento della riconnessione, i registri vengono inviati a CloudWatch Logs in the parent. Regione AWS Per monitorare il cluster in locale utilizzando l'endpoint dei parametri del server API Kubernetes o Fluent Bit per i log, puoi utilizzare Prometheus, Grafana o soluzioni partner di Amazon EKS.

- Se stai utilizzando un AWS Load Balancer Controller su Outposts per il traffico delle applicazioni, i AWS Load Balancer Controller esistenti anticipati dal Pods continueranno a ricevere traffico durante le disconnessioni dalla rete. I nuovi Pods creati durante le disconnessioni dalla rete non ricevono traffico fino a quando l'Outpost non ristabilisce la connessione con il Cloud AWS. Puoi impostare il numero di repliche per le tue applicazioni mentre sei connesso al Cloud AWS in base alle tue esigenze di dimensionamento durante le disconnessioni dalla rete.
- L'impostazione predefinita di Amazon VPC CNI plugin for Kubernetes è Modalità IP secondaria. È configurato con `WARM_ENI_TARGET=1`, che consente al plug-in di mantenere disponibile "un'interfaccia di rete completamente elastica" degli indirizzi IP disponibili. Puoi modificare i valori `WARM_ENI_TARGET`, `WARM_IP_TARGET` e `MINIMUM_IP_TARGET` in base alle tue esigenze di scalabilità durante uno stato di disconnessione. Per ulteriori informazioni, consultate il `readme`file relativo al plugin su. GitHub Per un elenco del numero massimo di elementi Pods supportati da ciascun tipo di istanza, consultate il `eni-max-pods.txt`file su GitHub.

Autenticazione al cluster locale durante una disconnessione dalla rete

AWS Identity and Access Management (IAM) non è disponibile durante le disconnessioni dalla rete. Non puoi autenticarti al cluster locale utilizzando le credenziali IAM durante una disconnessione. Tuttavia, durante la disconnessione, potrai connetterti al cluster tramite la rete locale utilizzando i certificati `x509`. È necessario scaricare e archiviare un certificato `X509` client da utilizzare durante le disconnessioni. In questo argomento viene illustrato come creare e utilizzare il certificato per autenticarsi al cluster quando è disconnesso.

1. Creare una richiesta di firma del certificato.

   a. Genera una richiesta di firma del certificato.

   ```
   openssl req -new -newkey rsa:4096 -nodes -days 365 \
       -keyout admin.key -out admin.csr -subj "/CN=admin"
   ```

   b. Crea una richiesta di firma del certificato in Kubernetes.

   ```
   BASE64_CSR=$(cat admin.csr | base64 -w 0)
   cat << EOF > admin-csr.yaml
   apiVersion: certificates.k8s.io/v1
   kind: CertificateSigningRequest
   metadata:
     name: admin-csr
   spec:
   ```

```
  signerName: kubernetes.io/kube-apiserver-client
  request: ${BASE64_CSR}
  usages:
  - client auth
EOF
```

2. Crea una richiesta di firma del certificato utilizzando `kubectl`.

```
kubectl create -f admin-csr.yaml
```

3. Controlla lo stato della richiesta di firma del certificato.

```
kubectl get csr admin-csr
```

Di seguito viene riportato un output di esempio:

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Pending
```

Kubernetes ha creato la richiesta di firma del certificato.

4. Approva la richiesta di firma del certificato.

```
kubectl certificate approve admin-csr
```

5. Controlla nuovamente lo stato della richiesta di firma del certificato per l'approvazione.

```
kubectl get csr admin-csr
```

Di seguito viene riportato un output di esempio:

```
NAME       AGE   REQUESTOR                      CONDITION
admin-csr  11m   kubernetes-admin               Approved
```

6. Recupera e verifica il certificato.

a. Recupera il certificato.

```
kubectl get csr admin-csr -o jsonpath='{.status.certificate}' | base64 --decode
 > admin.crt
```

b. Verifica il certificato.

```
cat admin.crt
```

7. Crea un'associazione di ruoli del cluster per un utente `admin`.

```
kubectl create clusterrolebinding admin --clusterrole=cluster-admin \
    --user=admin --group=system:masters
```

8. Genera un kubeconfig con ambito utente per uno stato disconnesso.

   Puoi generare un file `kubeconfig` utilizzando il certificati `admin` scaricati. Sostituisci *`my-cluster`* e *`apiserver-endpoint`* nei comandi seguenti.

```
aws eks describe-cluster --name my-cluster \
    --query "cluster.certificateAuthority" \
    --output text | base64 --decode > ca.crt
```

```
kubectl config --kubeconfig admin.kubeconfig set-cluster my-cluster \
    --certificate-authority=ca.crt --server apiserver-endpoint --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-credentials admin \
    --client-certificate=admin.crt --client-key=admin.key --embed-certs
```

```
kubectl config --kubeconfig admin.kubeconfig set-context admin@my-cluster \
    --cluster my-cluster --user admin
```

```
kubectl config --kubeconfig admin.kubeconfig use-context admin@my-cluster
```

9. Visualizza il file `kubeconfig`.

```
kubectl get nodes --kubeconfig admin.kubeconfig
```

10. Se disponi di servizi già in produzione sul tuo Outpost, salta questo passaggio. Se Amazon EKS è l'unico servizio in esecuzione sul tuo Outpost e l'Outpost non è attualmente in produzione, puoi simulare una disconnessione dalla rete. Prima di entrare in produzione con il cluster locale, puoi simulare una disconnessione per assicurarti di riuscire ad accedere al cluster quando è in uno stato disconnesso.

a. Applica le regole del firewall sui dispositivi di rete che connettono il tuo Outpost alla Regione AWS. Questo disconnette il link del servizio dell'Outpost. Non puoi creare nuove istanze. Le istanze attualmente in esecuzione perdono la connettività con la Regione AWS e Internet.

b. Puoi testare la connessione al cluster locale durante una disconnessione tramite il certificato `x509`. Assicurati di modificare la `kubeconfig` con il `admin.kubeconfig` che hai creato in una fase precedente. Sostituisci *`my-cluster`* con il nome del cluster locale.

```
kubectl config use-context admin@my-cluster --kubeconfig admin.kubeconfig
```

Se riscontri problemi con i cluster locali mentre sono disconnessi, ti consigliamo di inviare una richiesta di assistenza.

Considerazioni sulla capacità

Questo argomento fornisce assistenza per la selezione del tipo di istanza del piano di controllo (control-plane) Kubernetes e (facoltativamente) per l'uso di gruppi di collocamento per soddisfare requisiti di alta disponibilità per il tuo cluster Amazon EKS locale su un Outpost.

Prima di selezionare un tipo di istanza (`m5`, `c5` o `r5`) da utilizzare per il piano di controllo (control-plane) Kubernetes del tuo cluster locale su Outposts, controlla i tipi di istanza disponibili nella tua configurazione Outpost. Dopo aver identificato i tipi di istanza disponibili, devi selezionare la dimensione dell'istanza (`large`, `xlarge` o `2xlarge`) in base al numero di nodi richiesti dai carichi di lavoro. La tabella seguente fornisce consigli per la selezione della dimensione dell'istanza.

> **Note**
>
> Le dimensioni delle istanze devono essere assegnate ai tuoi Outposts. Assicurati di disporre di una capacità sufficiente per tre istanze della dimensione disponibile negli Outposts per tutta la durata del cluster locale. Per un elenco dei tipi di Amazon EC2 istanze disponibili, consulta le sezioni Calcolo e archiviazione nelle funzionalità del AWS Outposts rack.

Numero di nodi	Dimensione dell'istanza del piano di controllo (control-plane) Kubernetes
1-20	`large`
21-100	`xlarge`
101-250	`2xlarge`
251-500	`4xlarge`

L'archiviazione per il piano di controllo (control-plane) Kubernetes richiede 246 GB di spazio di archiviazione Amazon EBS per cluster locale per soddisfare gli IOPS richiesti da `etcd`. Quando viene creato il cluster, i volumi Amazon EBS vengono allocati automaticamente per tuo conto.

Collocamento del piano di controllo (control-plane)

Se non specifichi un gruppo di collocamento con la proprietà `OutpostConfig.ControlPlanePlacement.GroupName`, le istanze Amazon EC2 fornite per il tuo piano di controllo (control-plane) Kubernetes non ricevono alcuna specifica applicazione del collocamento dell'hardware nella capacità sottostante disponibile sul tuo Outpost.

Puoi utilizzare gruppi di collocamento per soddisfare i requisiti di alta disponibilità del tuo cluster Amazon EKS locale su un Outpost. Specificando un gruppo di collocamento durante la creazione del cluster, influisci sul collocamento delle istanze del piano di controllo (control-plane) Kubernetes. Le istanze sono distribuite su un hardware sottostante indipendente (rack o host), riducendo al minimo l'impatto di istanze correlate in caso di guasti hardware.

Requisiti

Il tipo di spread che puoi configurare dipende dal numero di rack Outpost esistenti nella tua implementazione.

- Implementazioni con uno o due rack fisici in un unico Outpost logico: occorrono almeno tre host configurati con il tipo di istanza scelto per le tue istanze del piano di controllo (control plane) Kubernetes. Un gruppo di collocamento spread che utilizza lo spread a livello di host garantisce che tutte le istanze del piano di controllo (control plane) Kubernetes vengano eseguite su host distinti all'interno dei rack sottostanti disponibili nella tua implementazione Outpost.

- Implementazioni con tre o più rack fisici in un unico Outpost logico: occorrono almeno tre host configurati con il tipo di istanza scelto per le tue istanze del piano di controllo (control plane) Kubernetes. Un gruppo di collocamento spread che utilizza lo spread a livello di host garantisce che tutte le istanze del piano di controllo (control plane) Kubernetes vengano eseguite su rack distinti nella tua implementazione Outpost. In alternativa, puoi utilizzare il gruppo di collocamento spread a livello di host come descritto nell'opzione precedente.

Sei responsabile della creazione del gruppo di collocamento desiderato. Specifichi il gruppo di collocamento quando richiami l'API `CreateCluster`. Per ulteriori informazioni sui gruppi di collocamento e su come crearli, consulta Placement Groups nella Amazon EC2 User Guide.

Considerazioni

- Quando viene specificato un gruppo di collocamento, deve essere disponibile una capacità slot sull'Outpost per creare correttamente un cluster Amazon EKS locale. La capacità varia a seconda se utilizzi il tipo di spread host o rack. Se la capacità è insufficiente, il cluster rimane nello stato `Creating`. Puoi controllare `Insufficient Capacity Error` nel campo di integrità della risposta dell'API `DescribeCluster`. Devi liberare capacità per l'avanzamento del processo di creazione.
- Durante gli aggiornamenti della piattaforma e della versione del cluster locale Amazon EKS, le istanze del piano di controllo (control plane) Kubernetes dal tuo cluster vengono sostituite da nuove istanze utilizzando una strategia di rotazione continua. Durante questo processo di sostituzione, ogni istanza del piano di controllo (control plane) termina, liberando il rispettivo slot. Al suo posto viene fornita una nuova istanza aggiornata. L'istanza aggiornata potrebbe essere collocata nello slot che è stato rilasciato. Se lo slot viene utilizzato da un'altra istanza non correlata e la capacità rimanente non soddisfa i requisiti di topologia spread richiesti, il cluster rimane nello stato `Updating`. Puoi controllare `Insufficient Capacity Error` rispettivo nel campo di integrità della risposta dell'API `DescribeCluster`. Devi liberare la capacità per l'avanzamento del processo di aggiornamento e la ridefinizione dei livelli di alta disponibilità precedenti.
- Puoi creare un massimo di 500 gruppi di collocamento per account ciascuno Regione AWS. Per ulteriori informazioni, consulta Regole e limitazioni generali nella Guida per l'utente di Amazon EC2.

Risoluzione dei problemi relativi ai cluster locali per Amazon EKS su AWS Outposts

Questo argomento illustra alcuni errori comuni che si potrebbero verificare durante l'utilizzo dei cluster locali e il modo in cui risolverli. I cluster locali siano simili ai cluster Amazon EKS sul cloud, ma esistono delle differenze nel modo in cui sono gestiti da Amazon EKS.

Funzionamento dell'API

I cluster locali vengono creati tramite l'API Amazon EKS, ma vengono eseguiti in modo asincrono. Ciò significa che le richieste all'API Amazon EKS vengono restituite immediatamente per i cluster locali. Tuttavia, queste richieste potrebbero avere esito positivo, anticipare l'errore a causa di errori di convalida degli input oppure fallire e riportare errori di convalida descrittivi. Questo funzionamento è simile a quello dell'API Kubernetes.

I cluster locali non effettuano la transizione a uno stato `FAILED`. Amazon EKS prova a riconciliare lo stato del cluster con lo stato desiderato richiesto dall'utente in modo continuo. Di conseguenza, un cluster locale può rimanere a lungo nello stato `CREATING`, fino a quando il problema di base non viene risolto.

Descrizione del campo di integrità del cluster

I problemi dei cluster locali possono essere rilevati con il comando `describe-cluster` della AWS CLI di Amazon EKS. I problemi relativi ai cluster locali vengono evidenziati dal campo `cluster.health` della risposta del comando `describe-cluster`. Il messaggio contenuto in questo campo include un codice di errore, un messaggio descrittivo e gli ID delle risorse correlate. Queste informazioni sono disponibili soltanto tramite l'API Amazon EKS e la AWS CLI. Nell'esempio seguente, sostituisci *`my-cluster`* con il nome del cluster locale.

```
aws eks describe-cluster --name my-cluster --query 'cluster.health'
```

Di seguito viene riportato un output di esempio:

```
{
    "issues": [
        {
            "code": "ConfigurationConflict",
            "message": "The instance type 'm5.large' is not supported in Outpost 'my-
outpost-arn'.",
            "resourceIds": [
```

```
                "my-cluster-arn"
            ]
        }
    ]
}
```

Se il problema non è risolvibile, potrebbe essere necessario eliminare il cluster locale e crearne uno nuovo. Ad esempio, ciò potrebbe verificarsi nel caso in cui si provi ad allocare un cluster con un tipo di istanza che non è disponibile sul tuo Outpost. La tabella seguente include gli errori più comuni relativi allo stato di integrità.

Scenario di errore	Codice	Messaggio	ResourceIds
Non è stato possibile trovare le sottoreti fornite.	`ResourceNotFound`	`The subnet ID subnet-id does not exist`	Tutti gli ID di sottorete forniti
Le sottoreti fornite non appartengono allo stesso VPC.	`ConfigurationConflict`	`Subnets specified must belong to the same VPC`	Tutti gli ID di sottorete forniti
Alcune delle sottoreti fornite non appartengono all'Outpost specificato.	`ConfigurationConflict`	`Subnet subnet-id expected to be in outpost-arn , but is in other-outpost-arn`	ID della sottorete problematico
Alcune delle sottoreti fornite non appartengono ad alcun Outpost.	`ConfigurationConflict`	`Subnet subnet-id is not part of any Outpost`	ID della sottorete problematico
Alcune sottoreti fornite non dispongono di indirizzi liberi sufficienti per creare interfacce di rete elastiche per	`ResourceLimitExceeded`	`The specified subnet does not have enough free addresses to satisfy the request.`	ID della sottorete problematico

Scenario di errore	Codice	Messaggio	ResourceIds
le istanze del piano di controllo.			
Il tipo di istanza del piano di controllo specificato non è supportato sul tuo Outpost.	`ConfigurationConflict`	`The instance type` *`type`* `is not supported in Outpost` *`outpost-arn`*	ARN del cluster
Hai terminato un'istanza Amazon EC2 del piano di controllo oppure `run-instance` ha avuto esito positivo ma sono state riscontrate modifiche a `Terminated` . Ciò può verificarsi per un periodo di tempo dopo la riconnessione dell'Outpost e gli errori interni di Amazon EBS causano il fallimento di un flusso di lavoro interno di Amazon EC2.	`InternalFailure`	`EC2 instance state "Terminated" is unexpected`	ARN del cluster

Scenario di errore	Codice	Messaggio	ResourceIds
Capacità insufficiente nell'Outpost. Ciò può accadere anche durante la creazione del cluster quando un Outpost viene disconnesso dalla Regione AWS.	`ResourceLimitExceeded`	`There is not enough capacity on the Outpost to launch or start the instance.`	ARN del cluster
Il tuo account supera la quota del gruppo di sicurezza	`ResourceLimitExceeded`	Messaggio di errore restituito dall'API Amazon EC2	ID del VPC di destinazione
Il tuo account supera la quota dell'interfaccia di rete elastica	`ResourceLimitExceeded`	Messaggio di errore restituito dall'API Amazon EC2	ID della sottorete di destinazione
Le istanze del piano di controllo non erano raggiungibili tramite AWS Systems Manager. Per la risoluzione, consulta la sezione Le istanze del piano di controllo non sono raggiungibili tramite AWS Systems Manager..	`ClusterUnreachable`	Le istanze del piano di controllo di Amazon EKS non sono raggiungibili tramite SSM. Verifica la configurazione SSM e di rete e fai riferimento alla documentazione per la risoluzione dei problemi di EKS su Outposts.	ID di istanza Amazon EC2

Scenario di errore	Codice	Messaggio	ResourceIds
Si è verificato un errore durante la raccolta dei dettagli per un gruppo di sicurezza gestito o un'interfaccia di rete elastica.	In base al codice di errore del client Amazon EC2.	Messaggio di errore restituito dall'API Amazon EC2	Tutti gli ID dei gruppi di sicurezza gestiti
Si è verificato un errore durante l'autorizzazione o la revoca delle regole di ingresso dei gruppi di sicurezza. Questo vale per i gruppi di sicurezza sia del cluster sia del piano di controllo.	In base al codice di errore del client Amazon EC2.	Messaggio di errore restituito dall'API Amazon EC2	ID gruppo di sicurezza problematico
Si è verificato un errore durante l'eliminazione di un'interfaccia di rete elastica per un'istanza del piano di controllo	In base al codice di errore del client Amazon EC2.	Messaggio di errore restituito dall'API Amazon EC2	ID dell'interfaccia di rete elastica problematica

Nella seguente tabella sono elencati gli errori di altri Servizi AWS che vengono presentati nel campo di integrità della risposta di `describe-cluster`:

Codice di errore Amazon EC2	Codice del problema di integrità del cluster	Descrizione
`AuthFailure`	`AccessDenied`	Questo problema può verificar si per una serie di motivi. Di solito si verificano se un

Codice di errore Amazon EC2	Codice del problema di integrità del cluster	Descrizione
		tag utilizzato dal servizio per definire la policy dei ruoli collegati al servizio viene rimosso accidentalmente dal piano di controllo. In tal caso, Amazon EKS non potrà più gestire e monitorare queste risorse AWS.
`UnauthorizedOperation`	`AccessDenied`	Questo problema può verificarsi per una serie di motivi. Di solito si verificano se un tag utilizzato dal servizio per definire la policy dei ruoli collegati al servizio viene rimosso accidentalmente dal piano di controllo. In tal caso, Amazon EKS non potrà più gestire e monitorare queste risorse AWS.
`InvalidSubnetID.NotFound`	`ResourceNotFound`	Questo errore si verifica quando non viene trovato l'ID di sottorete per le regole di ingresso di un gruppo di sicurezza.
`InvalidPermission.NotFound`	`ResourceNotFound`	Questo errore si verifica quando le autorizzazioni per le regole di ingresso di un gruppo di sicurezza non sono corrette.

Codice di errore Amazon EC2	Codice del problema di integrità del cluster	Descrizione
`InvalidGroup.NotFound`	`ResourceNotFound`	Questo errore si verifica quando non viene trovato il gruppo delle regole di ingresso di un gruppo di sicurezza.
`InvalidNetworkInterfaceID.NotFound`	`ResourceNotFound`	Questo errore si verifica quando non viene trovato l'ID dell'interfaccia di rete per le regole di ingresso di un gruppo di sicurezza.
`InsufficientFreeAddressesInSubnet`	`ResourceLimitExceeded`	Questo errore si verifica quando viene superata la quota di risorse della sottorete.
`InsufficientCapacityOnOutpost`	`ResourceLimitExceeded`	Questo errore si verifica quando viene superata la quota di capacità dell'outpost.
`NetworkInterfaceLimitExceeded`	`ResourceLimitExceeded`	Questo errore si verifica quando viene superata la quota dell'interfaccia di rete elastica.
`SecurityGroupLimitExceeded`	`ResourceLimitExceeded`	Questo errore si verifica quando viene superata la quota del gruppo di sicurezza.

Codice di errore Amazon EC2	Codice del problema di integrità del cluster	Descrizione
`VcpuLimitExceeded`	`ResourceLimitExceeded`	Ciò si verifica quando si crea un'istanza Amazon EC2 in un nuovo account. L'errore potrebbe essere simile al seguente: "`You have requested more vCPU capacity than your current vCPU limit of 32 allows for the instance bucket that the specified instance type belongs to. Please visit http://aws.amazon.com/contact-us/ec2-request to request an adjustment to this limit.`"
`InvalidParameterValue`	`ConfigurationConflict`	Amazon EC2 restituisce questo codice di errore se il tipo di istanza specificato non è supportato nell'Outpost.
Tutti gli altri errori	`InternalFailure`	Nessuno

Impossibile creare o modificare i cluster

I cluster locali richiedono autorizzazioni e policy diverse rispetto ai cluster Amazon EKS ospitati nel cloud. Quando un cluster non riesce a creare e produce un `InvalidPermissions` errore, ricontrolla che al ruolo del cluster che stai utilizzando sia associata la policy gestita di LocalOutpostClusterPolicy AmazonEks. Tutte le altre chiamate API richiedono lo stesso set di autorizzazioni dei cluster Amazon EKS sul cloud.

Il cluster è bloccato nello stato **`CREATING`**

La quantità di tempo necessaria per creare un cluster locale varia a seconda di diversi fattori. Questi fattori includono la configurazione della rete, la configurazione dell'Outpost e la configurazione del cluster. In generale, viene creato un cluster locale che passa allo stato `ACTIVE` entro 15-20 minuti. Se un cluster locale rimane nello stato `CREATING`, puoi richiamare `describe-cluster` per informazioni sulla causa nel campo di output `cluster.health`.

I problemi più comuni sono i seguenti:

AWS Systems Manager (Systems Manager) riscontra i seguenti problemi:

- Il cluster non può connettersi all'istanza del piano di controllo dalla Regione AWS in cui si trova Systems Manager. Puoi eseguire una verifica chiamando `aws ssm start-session --target instance-id` da un host bastione nella regione. Se il comando non funziona, controlla se Systems Manager è in esecuzione sull'istanza del piano di controllo. Oppure, un'altra soluzione alternativa consiste nell'eliminare il cluster e quindi ricrearlo.
- Le istanze del piano di controllo di Systems Manager potrebbero non avere accesso a Internet. Verifica se la sottorete fornita durante la creazione del cluster dispone di un gateway NAT e un VPC con un gateway Internet. Usa VPC Reachability Analyzer per verificare se l'istanza del control plane può raggiungere il gateway Internet. Per ulteriori informazioni, consulta la sezione Getting started with VPC Reachability Analyzer (Nozioni di base su VPC Reachability Analyzer).
- L'ARN del ruolo fornito è privo di alcune policy. Verifica che AWS politica gestita: AmazonEks LocalOutpostClusterPolicy sia stato rimosso dal ruolo. Ciò può verificarsi anche se uno stack AWS CloudFormation è configurato in modo errato.

Più sottoreti non sono state configurate correttamente e specificate durante la creazione di un cluster:

- Tutte le sottoreti fornite devono essere associate allo stesso Outpost ed essere in grado di raggiungersi tra loro. Quando vengono specificate più sottoreti durante la creazione del cluster, Amazon EKS prova a distribuire le istanze del piano di controllo su più sottoreti.
- I gruppi di sicurezza gestiti da Amazon EKS vengono applicati all'interfaccia di rete elastica. Tuttavia, altri elementi di configurazione, come le regole del firewall NACL, potrebbero essere in conflitto con le regole per l'interfaccia di rete elastica.

La configurazione del DNS del VPC e della sottorete non è configurata correttamente o è assente

Verificare Requisiti e considerazioni su VPC e sottoreti del cluster locale Amazon EKS.

Impossibile unire i nodi a un cluster

Cause comuni:

- Problemi relativi alle AMI:
  - Utilizzi un'AMI non supportata. Devi utilizzare v20220620 o versione successiva per l'AMI Amazon Linux ottimizzate per Amazon EKS Amazon Linux ottimizzata per Amazon EKS.
  - Se utilizzi un modello AWS CloudFormation per creare i tuoi nodi, assicurati che non utilizzi un'AMI non supportata.
- `ConfigMap` di AWS IAM Authenticator mancante: se manca, dovrai crearlo. Per ulteriori informazioni, consulta Applica la `aws-authConfigMap` al cluster.
- È stato utilizzato un gruppo di sicurezza non corretto: assicurati di utilizzare `eks-cluster-sg-`*`cluster-name`*`-`*`uniqueid`* per il gruppo di sicurezza dei nodi worker. Il gruppo di sicurezza selezionato viene modificato da AWS CloudFormation per consentire un nuovo gruppo di sicurezza ogni volta che lo stack viene utilizzato.
- A seguito dei passaggi del VPC di collegamento privato imprevisto: vengono passati dati CA (`--b64-cluster-ca`) o API Endpoint (`--apiserver-endpoint`) errati.
- Policy di sicurezza del Pod configurata in maniera errata:
  - I daemonset CoreDNS e Amazon VPC CNI plugin for Kubernetes devono essere eseguiti sui nodi affinché i nodi siano in grado di unirsi al cluster e comunicare correttamente con esso.
  - Per funzionare correttamente, il Amazon VPC CNI plugin for Kubernetes richiede alcune funzionalità di rete privilegiate. È possibile visualizzare le funzionalità di rete privilegiate con il comando seguente: `kubectl describe psp eks.privileged`.

  Ti consigliamo di modificare la policy di sicurezza del pod predefinita. Per ulteriori informazioni, consulta Policy di sicurezza pod.

Raccolta di registri

Quando un Outpost viene disconnesso dalla Regione AWS a cui è associato, è possibile che il cluster Kubernetes continui a funzionare normalmente. Tuttavia, se il cluster non funziona correttamente, segui la procedura di risoluzione dei problemi riportata in Preparazione alle disconnessioni dalla rete. Se riscontri altri problemi, contatta il AWS Support. AWS Support può guidarti nel download e nell'esecuzione di uno strumento di raccolta dei log. In questo modo, puoi raccogliere i log dalle istanze del piano di controllo del cluster Kubernetes e inviarli al AWS Support per ulteriori indagini.

Le istanze del piano di controllo non sono raggiungibili tramite AWS Systems Manager.

Quando le istanze del piano di controllo di Amazon EKS non sono raggiungibili tramite AWS Systems Manager (Systems Manager), Amazon EKS visualizza il seguente errore in relazione al tuo cluster.

```
Amazon EKS control plane instances are not reachable through SSM. Please verify your
 SSM and network configuration, and reference the EKS on Outposts troubleshooting
 documentation.
```

Per risolvere il problema, assicurati che il VPC e le sottoreti soddisfino i requisiti elencati in Requisiti e considerazioni su VPC e sottoreti del cluster locale Amazon EKS e di avere completato i passaggi riportati in Configurazione di Session Manager nella Guida per l'utente di AWS Systems Manager.

Avvio di nodi Amazon Linux autogestiti su un Outpost

In questo argomento viene descritto come avviare gruppi con scalabilità automatica di nodi Amazon Linux su un Outpost che si registrano con il cluster Amazon EKS. Il cluster può trovarsi su Cloud AWS o su un Outpost.

Prerequisiti

- Un Outpost esistente. Per ulteriori informazioni, consulta Cos'è AWS Outposts.
- Un cluster Amazon EKS esistente. Per distribuire un cluster su Cloud AWS, vedi. Creazione di un cluster Amazon EKS Per implementare un cluster su un Outpost, consulta la sezione Cluster locali per Amazon EKS su AWS Outposts.
- Supponiamo che tu stia creando i tuoi nodi in un cluster Cloud AWS e che tu abbia delle sottoreti nel luogo in Regione AWS cui li hai AWS Outposts o che AWS Local Zones sia AWS Wavelength abilitata. Quindi, le sottoreti non devono essere state passate al momento della creazione del cluster. Se stai creando i nodi in un cluster su un Outpost, durante la creazione del cluster devi avere passato una sottorete Outpost.
- (Consigliato per i cluster su Cloud AWS) Il Amazon VPC CNI plugin for Kubernetes componente aggiuntivo è configurato con il proprio ruolo IAM a cui è associata la politica IAM necessaria. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA). I cluster locali non supportano i ruoli IAM per gli account di servizio.

Puoi creare un gruppo di nodi Amazon Linux autogestito con `eksctl` o AWS Management Console (con un AWS CloudFormation modello). Puoi anche utilizzare Terraform.

eksctl

Prerequisito

La versione `0.183.0` o quelle successive dello strumento a riga di comando `eksctl` deve essere installata sul dispositivo o nella AWS CloudShell. Per l'installazione o l'aggiornamento di `eksctl`, consulta la sezione Installation nella documentazione di `eksctl`.

Avvio di nodi Linux autogestiti tramite **`eksctl`**

1. Se il cluster è sul Cloud AWS e la policy IAM gestita AmazonEKS_CNI_Policy è collegata a Ruolo IAM del nodo Amazon EKS, si consiglia, invece, di assegnarlo a un ruolo IAM associato all'account di servizio `aws-node` di Kubernetes. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA). Se il cluster è sull'Outpost, la policy deve essere collegata al ruolo del tuo nodo.
2. Il comando seguente crea un gruppo di nodi in un cluster esistente. Il cluster deve essere stato creato utilizzando `eksctl`. Sostituisci *`al-nodes`* con un nome per il gruppo di nodi. Il nome del gruppo di nodi non può contenere più di 63 caratteri. Deve iniziare con una lettera o un numero, ma può anche includere trattini e caratteri di sottolineatura. Sostituisci *`my-cluster`* con il nome del cluster. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può superare i 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster. Se il cluster si trova su un Outpost, sostituisci *`id`* con l'ID di una sottorete dell'Outpost. Se il cluster esiste su Cloud AWS, sostituiscilo *`id`* con l'ID di una sottorete che non hai specificato al momento della creazione del cluster. Sostituisci *`instance-type`* con un tipo di istanza supportato sul tuo Outpost. Sostituisci i *`example values`* rimanenti con i valori in tuo possesso. Per impostazione predefinita, i nodi vengono creati con la stessa versione Kubernetes del piano di controllo.

   Sostituisci *`instance-type`* con un tipo di istanza disponibile sul tuo Outpost.

   Sostituisci *`my-key`* con il nome della coppia di chiavi Amazon EC2 o della chiave pubblica. Questa chiave viene utilizzata per eseguire il SSH nei nodi dopo il loro avvio. Se non hai già una coppia di chiavi Amazon EC2, puoi crearla nella AWS Management Console. Per ulteriori informazioni, consulta la sezione relativa alle coppie di chiavi Amazon EC2 nella Guida per l'utente di Amazon EC2.

Crea il tuo gruppo di nodi con il comando seguente.

```
eksctl create nodegroup --cluster my-cluster --name al-nodes --node-
type instance-type \
    --nodes 3 --nodes-min 1 --nodes-max 4 --managed=false --node-volume-type gp2
 --subnet-ids subnet-id
```

Se il cluster è distribuito su: Cloud AWS

- Il gruppo di nodi implementato può assegnare gli indirizzi `IPv4` a Pods da un altro blocco CIDR rispetto a quello dell'istanza. Per ulteriori informazioni, consulta Rete personalizzata per i pod.
- Il gruppo di nodi che implementi non richiede l'accesso a Internet in uscita. Per ulteriori informazioni, consulta Requisiti dei cluster privati.

Per un elenco completo di tutte le impostazioni predefinite e le opzioni disponibili, consulta Supporto AWS Outposts nella documentazione di `eksctl`.

Se i nodi non riescono a unirsi al cluster, consulta le sezioni Impossibile aggiungere i nodi al cluster in Risoluzione dei problemi di Amazon EKS e Impossibile unire i nodi a un cluster in Risoluzione dei problemi relativi ai cluster locali per Amazon EKS su AWS Outposts.

Di seguito viene riportato un output di esempio: Durante la creazione dei nodi vengono generate diverse righe. Una delle ultime righe di output è simile alla seguente riga di esempio.

```
[#]  created 1 nodegroup(s) in cluster "my-cluster"
```

3. (Facoltativo) Implementa un'applicazione di esempio per testare il cluster e i nodi Linux.

AWS Management Console

Fase 1: Avviare nodi Amazon Linux autogestiti utilizzando il AWS Management Console

1. Scarica l'ultima versione del AWS CloudFormation modello.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-eks/cloudformation/2022-12-23/
amazon-eks-nodegroup.yaml
```

2. Apri la AWS CloudFormation console all'indirizzo https://console.aws.amazon.com/cloudformation.
3. Scegli Crea stack e quindi seleziona Con nuove risorse (standard).
4. In Specifica modello, seleziona Carica un file di modello e Scegli file. Seleziona il file `amazon-eks-nodegroup.yaml` scaricato in un passaggio precedente, quindi seleziona Next (Successivo).
5. Nella pagina Specifica i dettagli dello stack, immetti i parametri seguenti e scegli Successivo:

   - Nome stack: scegli il nome per lo stack di AWS CloudFormation . Ad esempio, è possibile chiamarlo ***`al-nodes`***. Il nome può contenere solo caratteri alfanumerici (con distinzione tra lettere maiuscole e minuscole) e trattini. Deve iniziare con un carattere alfanumerico e non può contenere più di 100 caratteri. Il nome deve essere univoco all'interno del Regione AWS e in Account AWS cui si sta creando il cluster.
   - ClusterName: inserisci il nome del cluster. Se questo nome non corrisponde al nome del cluster, i nodi non possono unirsi al cluster.
   - ClusterControlPlaneSecurityGruppo: scegli il SecurityGroupsvalore dall' AWS CloudFormation output che hai generato quando hai creato il tuo VPC.

     Nella procedura seguente viene illustrata un'operazione per recuperare il gruppo applicabile.

     1. Apri la console Amazon EKS all'indirizzo https://console.aws.amazon.com/eks/home#/clusters.
     2. Scegli il nome del cluster.
     3. Scegli la scheda Reti.
     4. Utilizza il valore dei gruppi di sicurezza aggiuntivi come riferimento quando selezioni dall'elenco a discesa ClusterControlPlaneSecurityGruppo.
   - NodeGroupNome: inserisci un nome per il tuo gruppo di nodi. Questo nome può essere utilizzato in seguito per identificare il gruppo di nodi con dimensionamento automatico creato per i tuoi nodi.
   - NodeAutoScalingGroupMinSize: Inserisci il numero minimo di nodi su cui il gruppo Auto Scaling del nodo può scalare.
   - NodeAutoScalingGroupDesiredCapacity: Inserisci il numero di nodi desiderato su cui scalare quando viene creato lo stack.
   - NodeAutoScalingGroupMaxSize: Inserisci il numero massimo di nodi su cui il gruppo Auto Scaling del nodo può scalare orizzontalmente.

- NodeInstanceTipo: scegli un tipo di istanza per i tuoi nodi. Se il cluster è in esecuzione su Cloud AWS, per ulteriori informazioni, consultaScelta di un tipo di istanza Amazon EC2. Se il cluster è in esecuzione su un Outpost, puoi selezionare solo un tipo di istanza disponibile sul tuo Outpost.
- NodeImageIDSSMParam: precompilato con il parametro Amazon EC2 Systems Manager di una recente AMI ottimizzata per Amazon EKS per una versione variabile. Kubernetes Per utilizzare una versione secondaria diversa di Kubernetes supportata da Amazon EKS, sostituisci *1.XX* con una versione supportata differente. Si consiglia di specificare la stessa versione Kubernetes del cluster.

  Per utilizzare l'AMI accelerata ottimizzata per Amazon EKS, sostituisci *amazon-linux-2* con **amazon-linux-2-gpu**. Per utilizzare l'AMI Arm ottimizzata per Amazon EKS, sostituisci *amazon-linux-2* con **amazon-linux-2-arm64**.

  > **Note**
  >
  > L'AMI del nodo Amazon EKS è basata su Amazon Linux. Tieni traccia degli eventi di sicurezza o di privacy per Amazon Linux 2 nel Centro di sicurezza di Amazon Linux o iscriviti al feed RSS associato. Gli eventi di sicurezza e privacy includono una panoramica del problema, quali sono i pacchetti interessati e come aggiornare le istanze per risolvere il problema.

- NodeImageID: (Facoltativo) Se utilizzi la tua AMI personalizzata (anziché l'AMI ottimizzata per Amazon EKS), inserisci un ID AMI del nodo per il tuo Regione AWS. Se specifichi un valore qui, questo sostituisce tutti i valori nel campo NodeImageIDSsmParam.
- NodeVolumeDimensione: specifica la dimensione del volume root per i tuoi nodi, in GiB.
- NodeVolumeTipo: specifica un tipo di volume root per i tuoi nodi.
- KeyName: inserisci il nome di una coppia di chiavi SSH Amazon EC2 che puoi usare per connetterti tramite SSH ai tuoi nodi dopo il loro avvio. Se non disponi di una coppia di chiavi Amazon EC2, puoi crearla nella AWS Management Console. Per ulteriori informazioni, consulta la sezione relativa alle coppie di chiavi Amazon EC2 nella Guida per l'utente di Amazon EC2.

> **Note**
>
> Se non fornisci una key pair qui, la creazione dello AWS CloudFormation stack fallisce.

- BootstrapArguments: Esistono diversi argomenti opzionali che puoi passare ai tuoi nodi. Per ulteriori informazioni, leggi le informazioni sull'utilizzo dello script di bootstrap su GitHub. Se stai aggiungendo nodi a un cluster locale Amazon EKS AWS Outposts (su cui vengono eseguite le istanze del piano di Kubernetes controllo AWS Outposts) e il cluster non dispone di una connessione Internet in ingresso e in uscita (nota anche come cluster privati), devi fornire i seguenti argomenti di bootstrap (come riga singola).

```
--b64-cluster-ca ${CLUSTER_CA} --apiserver-endpoint https://
${APISERVER_ENDPOINT} --enable-local-outpost true --cluster-id ${CLUSTER_ID}
```

- DisableIMDSv1: ogni nodo supporta Instance Metadat Service versione 1 (IMDSv1) e IMDSv2 per impostazione predefinita. Puoi disabilitare IMDSv1. Per evitare che in futuro i nodi e i Pods nel gruppo di nodi utilizzino IMDSv1, imposta DisableIMDSv1 su true. Per ulteriori informazioni su IMDS, consulta Configurazione del servizio di metadati dell'istanza. Per ulteriori informazioni sulle relative limitazioni dell'accesso ai nodi, consulta Limita l'accesso al profilo dell'istanza assegnato al nodo (worker).
- VpcId: inserisci l'ID per il VPC che hai creato. Prima di scegliere un VPC, consulta la sezione Considerazioni e requisiti relativi al VPC.
- Subnets: (Sottoreti) se il cluster è su un Outpost, scegli almeno una sottorete privata nel VPC. Prima di scegliere le sottoreti, consulta Considerazioni e requisiti relativi alla sottorete. È possibile visualizzare le sottoreti private aprendo ogni collegamento relativo alla sottorete dalla scheda Reti del cluster.

6. Seleziona le opzioni desiderate nella pagina Configura opzioni dello stack, quindi scegli Next (Avanti).
7. Seleziona la casella di controllo a sinistra di I acknowledge that AWS CloudFormation might create IAM resources (Riconosco che CFN potrebbe creare risorse IAM), quindi scegli Create stack (Crea stack).
8. Al termine della creazione dello stack, selezionalo nella console e scegli Output.
9. Registra il NodeInstanceruolo per il gruppo di nodi che è stato creato. Ciò sarà utile quando configurerai i nodi di Amazon EKS.

Fase 2: abilitazione dell'aggiunta di nodi al cluster

1. Verifica se disponi già di una `ConfigMap` per `aws-auth`.

```
kubectl describe configmap -n kube-system aws-auth
```

2. Se ti viene mostrata una `ConfigMap` per `aws-auth`, aggiornala se necessario.

   a. Apri `ConfigMap` per la modifica.

```
kubectl edit -n kube-system configmap/aws-auth
```

   b. Aggiungi una nuova voce `mapRoles`, se necessario. Impostate il `rolearn` valore sul valore del NodeInstanceruolo registrato nella procedura precedente.

```
[...]
data:
  mapRoles: |
    - rolearn: <ARN of instance role (not instance profile)>
      username: system:node:{{EC2PrivateDNSName}}
      groups:
        - system:bootstrappers
        - system:nodes
[...]
```

   c. Salva il file ed esci dall'editor di testo.

3. Se hai ricevuto un messaggio di errore che indica "`Error from server (NotFound): configmaps "aws-auth" not found`", applica lo `ConfigMap` di stock.

   a. Scarica la mappa di configurazione.

```
curl -O https://s3.us-west-2.amazonaws.com/amazon-
eks/cloudformation/2020-10-29/aws-auth-cm.yaml
```

   b. Nel `aws-auth-cm.yaml` file, impostate `rolearn` il valore NodeInstanceRole registrato nella procedura precedente. Per eseguire questa operazione, utilizza un editor di testo o sostituisci *`my-node-instance-role`* eseguendo il comando seguente:

```
sed -i.bak -e 's|<ARN of instance role (not instance profile)>|my-node-
instance-role|' aws-auth-cm.yaml
```

c. Applica la configurazione. L'esecuzione di questo comando potrebbe richiedere alcuni minuti.

```
kubectl apply -f aws-auth-cm.yaml
```

4. Guarda lo stato dei nodi e attendi che raggiungano lo stato `Ready`.

```
kubectl get nodes --watch
```

Inserisci `Ctrl+C` per tornare a un prompt dello shell (interprete di comandi).

> **Note**
>
> Se ricevi qualsiasi altro errore di tipo di risorsa o autorizzazione, consulta la sezione Accesso negato o non autorizzato (`kubectl`) nell'argomento relativo alla risoluzione dei problemi.

Se i nodi non riescono a unirsi al cluster, consulta le sezioni Impossibile aggiungere i nodi al cluster in Risoluzione dei problemi di Amazon EKS e Impossibile unire i nodi a un cluster in Risoluzione dei problemi relativi ai cluster locali per Amazon EKS su AWS Outposts.

5. Installa il driver CSI per Amazon EBS. Per ulteriori informazioni, vedere Installazione su GitHub. Nella sezione Set up driver permission (Configura autorizzazione del driver), assicurati di seguire le istruzioni per l'opzione Using IAM instance profile (Utilizzo del profilo dell'istanza IAM). È necessario utilizzare la classe di archiviazione `gp2`. La classe di archiviazione `gp3` non è supportata.

Per creare una classe di archiviazione `gp2` nel cluster, completa i seguenti passaggi.

1. Per creare il file `gp2-storage-class.yaml`, emetti il seguente comando:

```
cat >gp2-storage-class.yaml <<EOF
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  annotations:
    storageclass.kubernetes.io/is-default-class: "true"
  name: ebs-sc
provisioner: ebs.csi.aws.com
```

```
volumeBindingMode: WaitForFirstConsumer
parameters:
  type: gp2
  encrypted: "true"
allowVolumeExpansion: true
EOF
```

2. Applica il manifesto al cluster.

```
kubectl apply -f gp2-storage-class.yaml
```

6. (Solo nodi GPU) Se hai scelto un tipo di istanza GPU e l'AMI accelerata ottimizzata per Amazon EKS, devi applicare il plug-in del dispositivo NVIDIA per Kubernetes come DaemonSet sul cluster. Sostituisci *vX.X.X* con la versione plugin per dispositivi NVIDIA/K8Sdesiderata prima di eseguire il seguente comando.

```
kubectl apply -f https://raw.githubusercontent.com/NVIDIA/k8s-device-plugin/vX.X.X/nvidia-device-plugin.yml
```

Fase 3: operazioni aggiuntive

1. (Facoltativo) Implementa un'applicazione di esempio per testare il cluster e i nodi Linux.
2. Se il cluster è implementato su un Outpost, salta questo passaggio. Se il cluster è distribuito su Cloud AWS, le seguenti informazioni sono facoltative. Se la policy IAM gestita AmazonEKS_CNI_Policy è collegata al Ruolo IAM del nodo Amazon EKS, si consiglia, invece, di assegnarla a un ruolo IAM associato all'account di servizio Kubernetes `aws-node`. Per ulteriori informazioni, consulta Configurazione dell'Amazon VPC CNI plugin for Kubernetesutilizzo dei ruoli IAM per gli account di servizio (IRSA).

Progetti correlati

Questi progetti open source estendono la funzionalità dei cluster Kubernetes in esecuzione su o fuori AWS, inclusi i cluster gestiti da Amazon EKS.

Strumenti di gestione

Strumenti di gestione correlati per cluster Amazon EKS e Kubernetes.

eksctl

`eksctl` è un semplice strumento CLI per la creazione di cluster su Amazon EKS.

- URL progetto
- Documentazione progetto
- Blog open source di AWS: eksctl: cluster Amazon EKS con un comando

Controller AWS per Kubernetes

Con AWS Controllers per Kubernetes, puoi creare e gestire risorse AWS direttamente dal cluster Kubernetes.

- URL progetto
- Blog open source di AWS: Operatore del servizio AWS per Kubernetes ora disponibile

CD Flux

Flux è uno strumento utile per gestire la configurazione del cluster quando si usa Git. Utilizza un operatore nel cluster per attivare le implementazioni all'interno di Kubernetes. Per ulteriori informazioni sugli operatori, consulta OperatorHub.io su GitHub.

- URL progetto
- Documentazione progetto

CDK per Kubernetes

Con il CDK per Kubernetes (cdk8s), puoi definire app e componenti Kubernetes utilizzando linguaggi di programmazione familiari. Le app cdk8s si sintetizzano in manifesti Kubernetes standard, che possono essere applicati a qualsiasi cluster Kubernetes.

- URL progetto
- Documentazione progetto
- Blog container AWS: Introduzione di cdk8s+: API basate sugli intenti per oggetti Kubernetes

Rete

Progetti di rete correlati per cluster Amazon EKS e Kubernetes.

Amazon VPC CNI plugin for Kubernetes

Amazon EKS supporta reti VPC native tramite il Amazon VPC CNI plugin for Kubernetes. Il plug-in assegna un indirizzo IP dal VPC a ciascun Pod.

- URL progetto
- Documentazione progetto

AWS Load Balancer Controller per Kubernetes

Il AWS Load Balancer Controller consente di gestire gli Elastic Load Balancer di AWS per un cluster Kubernetes. Soddisfa le risorse di ingresso Kubernetes eseguendo il provisioning dell'Application Load Balancer AWS. Soddisfa le risorse del servizio Kubernetes eseguendo il provisioning di AWS Network Load Balancer.

- URL progetto
- Documentazione progetto

ExternalDNS

ExternalDNS sincronizza servizi e ingressi Kubernetes esposti con provider DNS, inclusi Amazon Route 53 e AWS Service Discovery.

- URL progetto
- Documentazione progetto

Machine learning

Progetti di machine learning correlati per cluster Amazon EKS e Kubernetes.

Kubeflow

Un kit di strumenti di machine learning per Kubernetes.

- URL progetto
- Documentazione progetto
- Blog open source di AWS: Kubeflow su Amazon EKS

Auto Scaling

Progetti di scalabilità automatica correlati per cluster Amazon EKS e Kubernetes.

Cluster Autoscaler

Cluster Autoscaler è uno strumento che consente di regolare automaticamente le dimensioni del cluster Kubernetes in base alle richieste di CPU e memoria.

- URL progetto
- Documentazione progetto
- Workshop Amazon EKS: https://www.eksworkshop.com/

Escalator

Escalator è un batch o autoscaler orizzontale ottimizzato per i processi per Kubernetes.

- URL progetto
- Documentazione progetto

Monitoraggio

Progetti di monitoraggio correlati per cluster Amazon EKS e Kubernetes.

Prometheus

Prometheus è un kit di strumenti di monitoraggio e avvisi di sistema open source.

- URL progetto
- Documentazione progetto
- Workshop Amazon EKS: https://eksworkshop.com/intermediate/240_monitoring/

Integrazione continua / distribuzione continua

Progetti CI/CD correlati per cluster Amazon EKS e Kubernetes.

Jenkins X

Soluzione CI/CD per applicazioni cloud moderne su cluster Amazon EKS e Kubernetes.

- URL progetto
- Documentazione progetto

Nuove funzionalità e roadmap di Amazon EKS

Per maggiori informazioni sulle nuove funzionalità di Amazon EKS, scorri fino al feed Novità nella pagina Cosa c'è di nuovo in AWS. È possibile anche rivedere la roadmap su GitHub, che consente di conoscere le funzionalità e le priorità in arrivo, in modo da poter pianificare come utilizzare Amazon EKS in futuro. È possibile fornirci un feedback diretto sulle priorità della roadmap.

Cronologia dei documenti per Amazon EKS

La tabella riportata di seguito illustra i principali aggiornamenti e le nuove caratteristiche della Guida per l'utente di Amazon EKS. Inoltre, aggiorniamo frequentemente la documentazione tenendo conto dei feedback ricevuti.

Modifica	Descrizione	Data
Kubernetes versione `1.30`	Sono stati aggiunti il supporto per Kubernetes versione `1.30` per nuovi cluster e alcuni aggiornamenti della versione.	23 maggio 2024
Aggiornamento della versione della piattaforma Amazon EKS	Si tratta di una nuova versione della piattaforma con correzioni della sicurezza e miglioramenti. Ciò include nuove versioni patch di Kubernetes e. `1.29.4` `1.28.9` `1.27.13`	14 maggio 2024
https://docs.aws.amazon.com/eks/latest/userguide/coredns-autoscaling.html	CoreDNSautoscaler adatterà dinamicamente il numero di repliche dell'CoreDNSimplementazione in un cluster EKS in base al numero di nodi e core della CPU. Questa funzionalità è valida sia per la versione di piattaforma più recente della versione di rilascio di EKS CoreDNS `v1.9` e successive. `1.25`	14 maggio 2024
CloudWatch Container Insights per Windows	Il Amazon CloudWatch Observability Operator componente aggiuntivo ora consente anche l'utilizzo	10 aprile 2024

	di Windows nodi di lavoro Container Insights nel cluster.	
Kubernetes	È stato aggiunto un nuovo argomento sui concetti di Kubernetes.	5 aprile 2024
Ristruttura l'accesso e i contenuti IAM	Sposta le pagine esistenti relative agli argomenti di accesso e IAM, come la mappa di configurazione di autenticazione, le voci di accesso, l'ID Pod e l'IRSA, in una nuova sezione. Rivedi il contenuto della panoramica.	2 aprile 2024
Supporto del Bottlerocket sistema operativo per il driver CSI Amazon S3	Il driver CSI Mountpoint for Amazon S3 è ora compatibile con. Bottlerocket	13 marzo 2024
AWS aggiornamenti delle politiche gestite: aggiornamento a una politica esistente	Amazon EKS ha aggiornato una policy AWS gestita esistente.	4 marzo 2024
2023	Amazon Linux 2023 (AL2023) è un nuovo sistema operativo basato su Linux progettato per fornire un ambiente sicuro, stabile e ad alte prestazioni per le tue applicazioni cloud.	29 febbraio 2024

EKS Pod Identity e IRSA supportano i sidecar in Kubernetes1.29	Nel Kubernetes1.29, i contenitori sidecar sono disponibili nei cluster Amazon EKS. I contenitori sidecar sono supportati con ruoli IAM per gli account di servizio o EKS Pod Identity. Per ulteriori informazioni sui sidecar, consulta Sidecar Containers nella documentazione. Kubernetes	26 febbraio 2024
Kubernetes versione 1.29	Sono stati aggiunti il supporto per Kubernetes versione 1.29 per nuovi cluster e alcuni aggiornamenti della versione.	23 gennaio 2024
Versione completa: Amazon EKS Extended Support per Kubernetes le versioni	Il supporto esteso per le versioni Kubernetes consente di utilizzare una versione Kubernetes specifica per più di 14 mesi.	16 gennaio 2024
Rilevamento dello stato di salute dei cluster Amazon AWS EKS nel cloud	Amazon EKS rileva problemi con i cluster Amazon EKS e l'infrastruttura dei prerequisiti del cluster nello stato del cluster. Puoi visualizzare i problemi con i tuoi cluster EKS all'interno AWS Management Console e all'interno `health` del cluster nell'API EKS. Questi problemi si aggiungono a quelli rilevati e visualizzati dalla console. In precedenza, lo stato del cluster era disponibile solo per i cluster locali su. AWS Outposts	28 dicembre 2023

di Amazon EKS Regione AWS	Amazon EKS è ora disponibile nella Regione AWS Canada occidentale (Calgary) (`ca-west-1`).	20 dicembre 2023
https://docs.aws.amazon.com/eks/latest/userguide/cluster-insights.html	Ora è possibile ottenere suggerimenti relativi al cluster sulla base di controlli ricorrenti.	20 dicembre 2023
Utilizzando le voci di accesso, ora è possibile concedere ai ruoli e agli utenti IAM l'accesso al cluster	Prima dell'introduzione delle voci di accesso, l'accesso al cluster veniva concesso ai ruoli e agli utenti IAM tramite l'aggiunta di voci a `aws-auth ConfigMap` . Adesso ogni cluster dispone di una modalità di accesso ed è possibile passare all'utilizzo di voci di accesso sulla base della propria pianificazione. Dopo aver cambiato modalità, puoi aggiungere utenti aggiungendo voci di accesso negli AWS CLI AWS CloudFormation, e negli AWS SDK.	18 dicembre 2023
Aggiornamento della versione della piattaforma Amazon EKS	Si tratta di una nuova versione della piattaforma con correzioni della sicurezza e miglioramenti. Ciò include nuove versioni di patch per Kubernetes `1.28.4`, `1.27.8`, `1.26.11` e `1.25.16`.	12 dicembre 2023

Driver CSI di Mountpoint per Amazon S3	Adesso puoi installare il driver CSI di Mountpoint per Amazon S3 sui cluster Amazon EKS.	27 novembre 2023
Attivazione dei parametri Prometheus durante la creazione di un cluster	In AWS Management Console, ora puoi attivare le Prometheus metriche durante la creazione di un cluster. Puoi anche visualizzare i dettagli dello scraper Prometheus nella scheda Osservabilità.	26 novembre 2023
Associazioni Amazon EKS Pod Identity	Le associazioni Amazon EKS Pod Identity associano un ruolo IAM a un account di servizio Kubernetes. Grazie a questa funzionalità, non è più necessario fornire autorizza zioni estese al ruolo IAM del nodo. In questo modo, Pods su quel nodo puoi chiamare le AWS API. A differenza dei ruoli IAM per gli account di servizio, le associazioni EKS Pod Identity sono completam ente interne a EKS; non è necessario un provider di identità OIDC.	26 novembre 2023
AWS aggiornamenti gestiti delle politiche: aggiornamento a una politica esistente	Amazon EKS ha aggiornat o una policy AWS gestita esistente.	26 novembre 2023

Controller di snapshot CSI	Ora puoi installare il controller di snapshot CSI per utilizzarlo con i driver CSI compatibili, come il driver CSI di Amazon EBS.	17 novembre 2023
Riscrittura dell'argomento Operatore ADOT	Il supporto del componente aggiuntivo di Amazon EKS per la sezione Operatore ADOT era ridondante con la documentazione AWS Distro per OpenTelemetry. Abbiamo migrato le informazioni essenziali rimanenti verso quella risorsa per ridurre le informazioni obsolete e incoerenti.	14 novembre 2023
Supporto per il componente aggiuntivo CoreDNS EKS per i parametri Prometheus	Le versioni `v1.10.1-eksbuild.5` , `v1.9.3-eksbuild.9` e `v1.8.7-eksbuild.8` del componente aggiuntivo EKS per CoreDNS espongono la porta su cui CoreDNS ha pubblicato i parametri, nel servizio `kube-dns`. Ciò semplifica l'inclusione dei parametri CoreDNS metriche nei sistemi di monitoraggio.	10 novembre 2023
Componente aggiuntivo Amazon EKS CloudWatch Observability Operator	Aggiunta la pagina Amazon EKS CloudWatch Observability Operator.	6 novembre 2023

Blocchi di capacità per istanze P5 autogestite nella Regione Stati Uniti orientali (Ohio)	Negli Stati Uniti orientali (Ohio), puoi ora utilizzare blocchi di capacità per istanze P5 autogestite.	31 ottobre 2023
I cluster supportano la modifica di sottoreti e gruppi di sicurezza	Puoi aggiornare il cluster per modificare le sottoreti e i gruppi di sicurezza da esso utilizzati. Puoi eseguire l' AWS Management Console aggiornamento dalla versione più recente di AWS CLI e dalla `eksctl` versione `v0.164.0-rc.0` successiva. AWS CloudFormation Potrebbe essere necessario eseguire questa operazione per fornire alle sottoreti più indirizzi IP disponibili per aggiornare in modo corretto una versione del cluster.	24 ottobre 2023

Il ruolo del cluster e il ruolo del gruppo di nodi gestiti supportano AWS Identity and Access Management le politiche gestite dai clienti	È possibile utilizzare una policy IAM personalizzata sul ruolo del cluster, anziché la policy `AmazonEKSClusterPolicy` AWS gestita. Inoltre, puoi utilizzare una policy IAM personalizzata sul ruolo del nodo in un gruppo di nodi gestiti, anziché la policy `AmazonEKSWorkerNodePolicy` AWS gestita. Esegui questa operazione per creare una policy con privilegi o minimo per soddisfare i rigorosi requisiti di conformità.	23 ottobre 2023
Correzione del collegamento per l'installazione di eksctl	Dopo lo spostamento della pagina, correggi il collegamento di installazione per eksctl.	6 ottobre 2023
Versione di anteprima: supporto esteso di Amazon EKS per le versioni Kubernetes	Il supporto esteso per le versioni Kubernetes consente di utilizzare una versione Kubernetes specifica per più di 14 mesi.	4 ottobre 2023
Rimuovi i riferimenti all' AWS App Mesh integrazione	Le integrazioni con Amazon EKS AWS App Mesh restano solo per i clienti esistenti di App Mesh.	29 settembre 2023
Kubernetes versione `1.28`	Sono stati aggiunti il supporto per Kubernetes versione `1.28` per nuovi cluster e alcuni aggiornamenti della versione.	26 settembre 2023

I cluster esistenti supportano l'applicazione delle policy Kubernetes di rete in Amazon VPC CNI plugin for Kubernetes	È possibile utilizzare la policy di rete Kubernetes nei cluster esistenti con il Amazon VPC CNI plugin for Kubernetes anziché richiedere una soluzione di terze parti.	15 settembre 2023
Il componente aggiuntivo CoreDNS di Amazon EKS supporta la modifica di PDB	Puoi modificare il componente aggiuntivo `PodDisruptionBudget` di EKS per CoreDNS nelle versioni `v1.9.3-eksbuild.7` e successive e `v1.10.1-eksbuild.4` e successive.	15 settembre 2023
Supporto Amazon EKS per sottoreti condivise	Nuova sezione Requisiti e considerazioni condivisi per le sottoreti per creare cluster Amazon EKS in sottoreti condivise.	7 settembre 2023
Aggiornamenti a Cos'è Amazon EKS?	Sono stati aggiunti nuovi argomenti nelle sezioni Casi d'uso comuni e Architettura. Altri argomenti sono stati aggiornati.	6 settembre 2023
Applicazione delle policy di rete di Kubernetes nel Amazon VPC CNI plugin for Kubernetes	È possibile utilizzare la policy di rete Kubernetes con il Amazon VPC CNI plugin for Kubernetes anziché richiedere una soluzione di terze parti.	29 agosto 2023
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile in Israele (Tel Aviv) (`il-central-1`) Regione AWS.	1° agosto 2023

Archiviazione temporanea configurabile per Fargate	È possibile aumentare la quantità totale di spazio di archiviazione effimero per ciascun Pod in esecuzione su Amazon EKS Fargate.	31 luglio 2023
Supporto aggiuntivo per il driver CSI per Amazon EBS	Ora puoi utilizzare l'API AWS Management Console AWS CLI, e per gestire il driver CSI di Amazon EFS.	26 luglio 2023
AWS aggiornamenti delle politiche gestite: nuova politica	Amazon EKS ha aggiunto una nuova policy AWS gestita.	26 luglio 2023
Kubernetesgli aggiornamenti delle versioni per 1.27, 1.26, 1.25 e 1.24 sono ora disponibili per i cluster locali su AWS Outposts	Kubernetesgli aggiornamenti delle versioni 1.27.3, 1.26.6, 1.25.11 e 1.24.15 sono ora disponibili per i cluster locali su AWS Outposts	20 luglio 2023
Supporto dei prefissi IP per nodi Windows	L'assegnazione di prefissi IP ai nodi può consentirti di ospitare un numero di gran lunga maggiore di Pods sui tuoi nodi rispetto a quanti puoi ospitarne quando assegni singoli indirizzi IP secondari ai tuoi nodi.	6 luglio 2023
Driver CSI per Amazon FSx per OpenZFS	Adesso puoi installare il driver CSI di Amazon FSx per OpenZFS sui cluster Amazon EKS.	30 giugno 2023

Pods sui nodi Linux nei cluster IPv4 ora possono comunicare con gli endpoint IPv6.	Dopo aver assegnato un indirizzo IPv6 al tuo nodo, il tuo indirizzo Pods' IPv4 è l'indirizzo di rete tradotto nell'indirizzo IPv6 del nodo su cui è in esecuzione.	19 giugno 2023
Windowsgruppi di nodi gestiti in AWS GovCloud (US) Regions	In AWS GovCloud (US) Regions, i gruppi di nodi gestiti di Amazon EKS possono ora eseguire Windows contenitori.	30 maggio 2023
Kubernetes versione 1.27	Sono stati aggiunti il supporto per Kubernetes versione 1.27 per nuovi cluster e alcuni aggiornamenti della versione.	24 maggio 2023
Kubernetes versione 1.26	Sono stati aggiunti il supporto per Kubernetes versione 1.26 per nuovi cluster e alcuni aggiornamenti della versione.	11 aprile 2023
gMSA senza dominio	Ora puoi utilizzare gMSA senza dominio con i Pods Windows.	27 marzo 2023
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nella regione Asia-Pacifico (Melbourne) (ap-southeast-4) Regione AWS.	10 marzo 2023
Driver CSI di Amazon File Cache	Ora è possibile installare il driver CSI per Amazon File Cache sui cluster Amazon EKS.	3 marzo 2023

Kubernetesla versione 1.25 è ora disponibile per i cluster locali su AWS Outposts	Ora puoi creare un cluster locale Amazon EKS su un Outpost utilizzando le versioni da `1.22` a `1.25` di Kubernetes.	1 marzo 2023
Kubernetes versione `1.25`	Sono stati aggiunti il supporto per Kubernetes versione `1.25` per nuovi cluster e alcuni aggiornamenti della versione.	22 febbraio 2023
AWS aggiornamenti delle politiche gestite: aggiornamento a una politica esistente	Amazon EKS ha aggiornato una policy AWS gestita esistente.	7 febbraio 2023
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile in Asia Pacifico (Hyderabad) (`ap-south-2`), Europa (Zurigo) (`eu-central-2`) ed Europa (Spagna) (`eu-south-2`). Regioni AWS	6 febbraio 2023
Kubernetesversioni`1.21`: ora `1.24` sono disponibili per i cluster locali su. AWS Outposts	Ora puoi creare un cluster locale Amazon EKS su un Outpost utilizzando le versioni da `1.21` a `1.24` di Kubernetes. In precedenza, era disponibile solo la versione `1.21`.	17 gennaio 2023
Amazon EKS ora supporta AWS PrivateLink	Puoi usare an AWS PrivateLink per creare una connessione privata tra il tuo VPC e Amazon EKS.	16 dicembre 2022
Supporto Windows per gruppi di nodi gestiti	Ora puoi utilizzare Windows per gruppi di nodi gestiti Amazon EKS.	15 dicembre 2022

Ora sono disponibili componenti aggiuntivi di Amazon EKS di fornitori di software indipendenti in Marketplace AWS	Ora puoi individuare e sottoscrivere componenti aggiuntivi di Amazon EKS di fornitori di software indipendenti tramite Marketplace AWS.	28 novembre 2022
AWS aggiornamenti gestiti delle politiche: aggiornamento a una politica esistente	Amazon EKS ha aggiornato una policy AWS gestita esistente.	17 novembre 2022
Kubernetes versione `1.24`	Sono stati aggiunti il supporto per Kubernetes versione `1.24` per nuovi cluster e alcuni aggiornamenti della versione.	15 novembre 2022
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile in Medio Oriente (Emirati Arabi Uniti) (`me-central-1`). Regione AWS	3 novembre 2022
AWS aggiornamenti gestiti delle politiche: aggiornamento a una politica esistente	Amazon EKS ha aggiornato una policy AWS gestita esistente.	24 ottobre 2022
AWS aggiornamenti gestiti delle politiche: aggiornamento a una politica esistente	Amazon EKS ha aggiornato una policy AWS gestita esistente.	20 ottobre 2022
I cluster locali attivi AWS Outposts sono ora disponibili	Ora puoi creare un cluster locale Amazon EKS su un Outpost.	19 settembre 2022
Quote basate su vCPU Fargate	Fargate sta passando dalle quote basate su Pod alle quote basate su vCPU.	8 settembre 2022

AWS aggiornamenti gestiti delle politiche: aggiornamento a una politica esistente	Amazon EKS ha aggiornato una policy AWS gestita esistente.	31 agosto 2022
Monitoraggio dei costi	Amazon EKS supporta ora Kubecost, che consente di monitorare i costi suddivisi per risorse Kubernetes tra cui Pods, nodi, spazi dei nomi ed etichette.	24 agosto 2022
AWS aggiornamenti delle politiche gestite: nuova politica	Amazon EKS ha aggiunto una nuova policy AWS gestita.	24 agosto 2022
AWS aggiornamenti delle politiche gestite: nuova politica	Amazon EKS ha aggiunto una nuova policy AWS gestita.	23 agosto 2022
Aggiunta di tag alle risorse per la fatturazione	Aggiunto il supporto `aws:eks:cluster-name` generato per i tag di allocazione dei costi per tutti i cluster.	16 agosto 2022
Caratteri jolly del profilo Fargate	È stato aggiunto il supporto per i caratteri jolly del profilo Fargate nei criteri di selezione per gli spazi dei nomi, le chiavi e i valori delle etichette.	16 agosto 2022
Kubernetes versione `1.23`	Sono stati aggiunti il supporto per Kubernetes versione `1.23` per nuovi cluster e alcuni aggiornamenti della versione.	11 agosto 2022
Visualizza Kubernetes le risorse in AWS Management Console	Ora puoi visualizzare le informazioni relative alle risorse Kubernetes implementate nel cluster utilizzando la AWS Management Console.	3 maggio 2022

Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nella regione Asia-Pacifico (Giacarta) (`ap-southeast-3`). Regione AWS	2 maggio 2022
Pagina Osservabilità e supporto per il componente aggiuntivo ADOT	Aggiunte la pagina Observability e AWS Distro for OpenTelemetry (ADOT).	21 aprile 2022
Kubernetes versione `1.22`	Sono stati aggiunti il supporto per Kubernetes versione `1.22` per nuovi cluster e alcuni aggiornamenti della versione.	4 aprile 2022
AWS aggiornamenti delle politiche gestite - Nuova politica	Amazon EKS ha aggiunto una nuova policy AWS gestita.	4 aprile 2022
Aggiunti i dettagli relativi all'applicazione di patch ai Pod Fargate	Durante l'aggiornamento dei Pods Fargate, Amazon EKS prova innanzitutto a espellere i Pods in base ai relativi budget di interruzione dei Pod. È possibile creare regole di eventi per contrastare le espulsioni non riuscite prima che i Pods vengano eliminati.	1 aprile 2022
Versione completa: supporto aggiuntivo per il driver CSI di Amazon EBS	Ora puoi utilizzare l'API AWS Management Console AWS CLI, e per gestire il driver CSI di Amazon EBS.	31 marzo 2022
AWS Outposts aggiornamento dei contenuti	Istruzioni per l'implementazione di un cluster Amazon EKS in AWS Outposts.	22 marzo 2022

AWS aggiornamenti delle politiche gestite: aggiornamento a una politica esistente	Amazon EKS ha aggiornato una policy AWS gestita esistente.	21 marzo 2022
Supporto `containerd` Windows	Ora puoi selezionare il runtime `containerd` per i nodi Windows.	14 marzo 2022
Sono state aggiunte alcune considerazioni in merito ad Amazon EKS Connector alla documentazione di sicurezza	Descrive il modello di responsabilità condivisa in relazione ai cluster connessi.	25 febbraio 2022
Assegnazione degli indirizzi `IPv6` ai Pods e ai servizi	Ora puoi creare un cluster `1.21` o versione successiva che assegna gli indirizzi `IPv6` a Pods e servizi.	6 gennaio 2022
AWS aggiornamenti gestiti delle politiche: aggiornamento a una politica esistente	Amazon EKS ha aggiornato una policy AWS gestita esistente.	13 dicembre 2021
Versione di anteprima: supporto del componente aggiuntivo del driver CSI per Amazon EBS	Ora puoi visualizzare l'anteprima utilizzando l'API AWS Management Console AWS CLI, e per gestire il driver CSI di Amazon EBS.	9 dicembre 2021
Supporto autoscaler Karpenter	Ora puoi utilizzare il progetto open source Karpenter per scalare automaticamente i nodi.	29 novembre 2021
Supporto filtro Fluent Bit Kubernetes nella registrazione di Fargate	Ora puoi utilizzare il filtro Kubernetes Fluent Bit con il logging di Fargate.	10 novembre 2021

Supporto Windows disponibile nel piano di controllo	Il supporto Windows è ora disponibile nel piano di controllo. Non è più necessario attivarlo nel piano dati.	9 novembre 2021
Bottlerocket aggiunto come tipo di AMI per i gruppi di nodi gestiti	In precedenza, Bottlerocket era disponibile solo come opzione di nodo autogestito. Ora può essere configurato come gruppo di nodi gestiti, riducendo lo sforzo necessari o per soddisfare i requisiti di conformità dei nodi.	28 ottobre 2021
Supporto driver DL1	Le AMI Amazon Linux personalizzate ora supportan o i carichi di lavoro di deep learning per Amazon Linux 2. Questa abilitazione consente una configurazione generica on-premise o di base su cloud.	25 ottobre 2021
Supporto video VT1	Le AMI personalizzate Amazon Linux ora supportan o VT1 per alcune distribuzioni. Questa abilitazione pubbliciz za i dispositivi Xilinx U30 sul cluster Amazon EKS.	13 settembre 2021
È ora disponibile Amazon EKS Connector	Puoi utilizzare Amazon EKS Connector per registrar e e connettere qualsiasi Kubernetes cluster conforme AWS e visualizzarlo nella console Amazon EKS.	8 settembre 2021

È ora disponibile Amazon EKS Anywhere	Amazon EKS Anywhere è una nuova opzione di implement azione per Amazon EKS che consente di creare e gestire cluster Kubernetes on-premis e.	8 settembre 2021
Driver CSI Amazon FSx per NetApp ONTAP	È stato aggiunto un argomento che riassume il driver CSI Amazon FSx NetApp for ONTAP e fornisce collegame nti ad altri riferimenti.	2 settembre 2021
I gruppi di nodi gestiti ora calcolano automaticamente il numero massimo di Pods consigliati da Amazon EKS per i nodi	Ora i gruppi di nodi gestiti calcolano in automatico il numero massimo di Pods Amazon EKS per i nodi che implementi senza un modello di avvio o con un modello di avvio in cui non hai specificato un ID AMI.	30 agosto 2021
Rimozione della gestione delle impostazioni dei componenti aggiuntivi da parte di Amazon EKS, senza rimuovere il software aggiuntivo Amazon EKS	Ora è possibile rimuovere un componente aggiuntivo Amazon EKS senza rimuovere il software aggiuntivo dal cluster.	20 agosto 2021
Creazione di Pods multihomed con Multus	Ora puoi aggiungere più interfacce di rete a un Pod utilizzando Multus.	2 agosto 2021

Aggiunta di altri indirizzi IP ai nodi Linux Amazon EC2	È ora possibile aggiungere significativamente più indirizzi IP ai nodi Linux Amazon EC2. Ciò significa che puoi eseguire una maggiore densità di Pods su ciascun nodo.	27 luglio 2021
`containerd` bootstrap del runtime	L'Amazon Machine Image (AMI) di Amazon Linux ottimizzata e accelerata per Amazon EKS ora contiene un flag di bootstrap per abilitare facoltativamente il runtime `containerd` in Amazon EKS ottimizzato e le AMI Bottlerocket. Questo flag è disponibile in tutte le versioni Kubernetes supportat e dell'AMI.	19 luglio 2021
Kubernetes versione `1.21`	È stato aggiunto il supporto per la versione `1.21` diKubernetes.	19 luglio 2021
É stato aggiunto l'argomento relativo alle policy gestite	Un elenco di tutte le policy IAM gestite Amazon EKS e le modifiche apportate loro dal 17 giugno 2021.	17 giugno 2021
Utilizzo dei gruppi di sicurezza per i Pods con Fargate	Ora puoi utilizzare i gruppi di sicurezza per i Pods con Fargate oltre che con i nodi Amazon EC2.	1 giugno 2021

Sono stati aggiunti i componenti aggiuntivi CoreDNS e `kube-proxy` Amazon EKS	Amazon EKS può ora aiutarti a gestire i componenti aggiuntivi CoreDNS e `kube-proxy` di Amazon EKS per il cluster.	19 maggio 2021
Kubernetes versione `1.20`	Sono stati aggiunti il supporto per Kubernetes versione `1.20` per nuovi cluster e alcuni aggiornamenti della versione.	18 maggio 2021
È stato rilasciato AWS Load Balancer Controller`2.2.0`	È ora possibile utilizzare il AWS Load Balancer Controller per creare Elastic Load Balancer utilizzando destinazioni istanza o IP.	14 maggio 2021
Contaminazioni di nodi per i gruppi di nodi gestiti	Amazon EKS ora supporta l'aggiunta di note di contaminazione ai gruppi di nodi gestiti.	11 maggio 2021
Crittografia dei segreti per i cluster esistenti	Amazon EKS ora supporta l'aggiunta della crittografia dei segreti per i cluster esistenti.	26 febbraio 2021
Kubernetes versione `1.19`	Sono stati aggiunti il supporto per Kubernetes versione `1.19` per nuovi cluster e alcuni aggiornamenti della versione.	16 febbraio 2021
Amazon EKS ora supporta i provider di identità OpenID Connect (OIDC) come metodo per autenticare gli utenti nel cluster aggiornato alla versione `1.16` o successiva.	I gestori dell'identità digitale OIDC possono essere utilizzati con (o in alternativa a) AWS Identity and Access Management (IAM).	12 febbraio 2021

Visualizza le risorse dei nodi e dei carichi di lavoro nel AWS Management Console	Ora è possibile visualizzare i dettagli sui nodi gestiti, autogestiti e Fargate e sui carichi di lavoro Kubernetes implementati nella AWS Management Console.	1 dicembre 2020
Implementazione dei tipi di istanza spot in un gruppo di nodi gestiti	È ora possibile implementare più tipi di istanza Spot o On-Demand a un gruppo di nodi gestito.	1 dicembre 2020
Ora Amazon EKS può gestire componenti aggiuntivi specifici per il tuo cluster	Puoi gestire i componenti aggiuntivi in autonomia o consentire ad Amazon EKS di controllare l'avvio e la versione di un componente aggiuntivo tramite l'API Amazon EKS.	1 dicembre 2020
Condivisione di un ALB su più ingressi	Ora puoi condividere un AWS Application Load Balancer (ALB) tra più ingressi. Kubernetes In passato, era necessario implementare un ALB separato per ogni ingresso.	23 ottobre 2020
Supporto destinazioni IP per NLB	Ora è possibile implementare un Network Load Balancer con destinazioni IP. Ciò significa che puoi utilizzare un Network Load Balancer per bilanciare il traffico di rete verso i Pods Fargate e direttamente verso i Pods in esecuzione sui nodi Amazon EC2.	23 ottobre 2020

Kubernetes versione 1.18	Sono stati aggiunti il supporto per Kubernetes versione 1.18 per nuovi cluster e alcuni aggiornamenti della versione.	13 ottobre 2020
Specificare un blocco CIDR personalizzato per l'assegnazione dell'indirizzo IP del servizio Kubernetes.	Ora è possibile specificare un blocco CIDR personalizzato a cui Kubernetes assegna gli indirizzi IP del servizio.	29 settembre 2020
Assegnazione dei gruppi di sicurezza ai singoli Pods	Ora è possibile associare diversi gruppi di sicurezza ad alcuni dei singoli Pods in esecuzione su molti tipi di istanza Amazon EC2.	9 settembre 2020
Implementazione di Bottlerocket sui nodi	Ora è possibile implementare i nodi che eseguono Bottlerocket.	31 agosto 2020
La possibilità di avviare i nodi Arm è generalmente disponibile	È possibile avviare i nodi Arm in gruppi di nodi gestiti e autogestiti.	17 agosto 2020
Modelli di avvio di gruppi di nodi gestiti e AMI personalizzate	È possibile implementare un gruppo di nodi gestito che utilizza un modello di avvio di Amazon EC2. Se lo si preferisce, il modello di avvio può specificare un'AMI personalizzata.	17 agosto 2020
Supporto EFS per AWS Fargate	Ora puoi usare Amazon EFS con AWS Fargate.	17 agosto 2020

Aggiornamento della versione della piattaforma Amazon EKS	Si tratta di una nuova versione della piattaforma con correzioni della sicurezza e miglioramenti. È incluso il supporto UDP per i servizi di tipo `LoadBalancer` quando si utilizzano Network Load Balancer con Kubernetes versione `1.15` o successiva. Per ulteriori informazioni, consulta il problema Allow UDP for AWS Network Load Balancer su. GitHub	12 agosto 2020
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nelle Regioni AWS Africa (Città del Capo) (`af-south-1`) ed Europa (Milano) (`eu-south-1`).	6 agosto 2020
Parametri di utilizzo Fargate	AWS Fargate fornisce metriche di CloudWatch utilizzo che forniscono visibilità sull'utilizzo delle risorse Fargate On-Demand da parte dell'account.	3 agosto 2020
Kubernetes versione `1.17`	Sono stati aggiunti il supporto per Kubernetes versione `1.17` per nuovi cluster e alcuni aggiornamenti della versione.	10 luglio 2020

Creazione e gestione delle risorse App Mesh da Kubernetes con il controller App Mesh per Kubernetes	Puoi creare e gestire risorse App Mesh dall'interno di Kubernetes. Il controller inserisce in automatico il proxy Envoy e i container init nei Pods implementati.	18 giugno 2020
Amazon EKS ora supporta i nodi Inf1 Amazon EC2	É possibile aggiungere nodi Inf1 Amazon EC2 al cluster.	4 giugno 2020
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile negli AWS GovCloud (Stati Uniti orientali) (`us-gov-east-1`) e AWS GovCloud (Stati Uniti occidentali) (`us-gov-west-1`). Regioni AWS	13 maggio 2020
Kubernetes`1.12` non è più supportato in Amazon EKS	Kubernetes versione `1.12` non è più supportato in Amazon EKS. Esegui l'aggiornamento di eventuali cluster `1.12` alla versione `1.13` o successiva per evitare interruzioni del servizio.	12 maggio 2020
Kubernetes versione `1.16`	Sono stati aggiunti il supporto per Kubernetes versione `1.16` per nuovi cluster e alcuni aggiornamenti della versione.	30 aprile 2020
È stato aggiunto il ruolo collegato al servizio AWSServiceRoleForAmazonEKS	È stato aggiunto il ruolo collegato al AWSServiceRoleForAmazonEKSservizio.	16 aprile 2020

Kubernetes versione 1.15	Sono stati aggiunti il supporto per Kubernetes versione 1.15 per nuovi cluster e alcuni aggiornamenti della versione.	10 marzo 2020
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nelle Regioni AWS Pechino (cn-north-1) e Ningxia (cn-northwest-1).	26 febbraio 2020
Driver CSI per Amazon FSx per Lustre	Aggiunto l'argomento relativo all'installazione del driver CSI per FSx per Lustre nei cluster Amazon EKS con Kubernetes 1.14.	23 dicembre 2019
Limitare l'accesso di rete all'endpoint di accesso pubblico di un cluster	Questo aggiornamento consente ad Amazon EKS di limitare gli intervalli CIDR che possono comunicare con l'endpoint di accesso pubblico del server API Kubernetes.	20 dicembre 2019
Risolvere l'indirizzo dell'endpoint di accesso privato per un cluster dall'esterno di un VPC	Con questo aggiornamento, è possibile utilizzare Amazon EKS per risolvere l'endpoint di accesso privato del server API Kubernetes dall'esterno di un VPC.	13 dicembre 2019
(Beta) Nodi di istanza Amazon EC2 A1 di Amazon EC2	Avvio di nodi di istanza Amazon EC2 A1 di Amazon EC2 che si registrano con il cluster Amazon EKS.	4 dicembre 2019
Creare un cluster su AWS Outposts	Amazon EKS ora supporta la creazione di cluster in AWS Outposts.	3 dicembre 2019

AWS Fargate su Amazon EKS	Ora i cluster Kubernetes di Amazon EKS supportano i Pods in esecuzione su Fargate.	3 dicembre 2019
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile in Canada (Central) (`ca-central-1`) Regione AWS.	21 novembre 2019
Gruppi di nodi gestiti	I gruppi di nodi gestiti Amazon EKS automatizzano il provisioning e la gestione del ciclo di vita dei nodi (istanze Amazon EC2) per i cluster Kubernetes Amazon EKS.	18 novembre 2019
Aggiornamento della versione della piattaforma Amazon EKS	Nuove versioni della piattaforma per gestire `CVE-2019-11253`.	6 novembre 2019
Kubernetes`1.11` non è più supportato in Amazon EKS	Kubernetes versione `1.11` non è più supportato in Amazon EKS. Esegui l'aggiornamento di eventuali cluster `1.11` alla versione `1.12` o successiva per evitare interruzioni del servizio.	4 novembre 2019
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nella Regione AWS Sud America (San Paolo) (`sa-east-1`).	16 ottobre 2019
Supporto Windows	I cluster Amazon EKS che eseguono Kubernetes versione `1.14` ora supportano i carichi di lavoro di Windows.	7 ottobre 2019

Dimensionamento automatico	È stato aggiunto il capitolo per descrivere alcuni dei vari tipi di scalabilità automatic a Kubernetes supportati nei cluster Amazon EKS.	30 settembre 2019
Aggiornamento della dashboard Kubernetes	È stato aggiornato l'argomen to relativo all'installazione del pannello di controllo Kubernetes nei cluster Amazon EKS per l'uso della versione beta `2.0`.	28 settembre 2019
Driver CSI per Amazon EFS	Aggiunto l'argomento relativo all'installazione del driver CSI per Amazon EFS nei cluster Amazon EKS con Kubernetes `1.14`.	19 settembre 2019
Parametro di Amazon EC2 Systems Manager per ID di AMI ottimizzate per Amazon EKS	É stato aggiunto l'argomen to relativo al recupero dell'ID AMI ottimizzata per Amazon EKS mediante un parametro Amazon EC2 Systems Manager. Il parametro elimina la necessità di cercare gli ID AMI.	18 settembre 2019
Assegnazione di tag alle risorse di Amazon EKS	È possibile gestire l'assegna zione di tag ai propri cluster Amazon EKS.	16 settembre 2019
Driver CSI per Amazon EBS	Aggiunto l'argomento relativo all'installazione del driver CSI per Amazon EBS nei cluster Amazon EKS con Kubernetes `1.14`.	9 settembre 2019

Nuova AMI ottimizzata per Amazon EKS con patch per CVE-2019-9512 e CVE-2019-9514	Amazon EKS ha aggiornato l'AMI ottimizzata per Amazon EKS per la gestione di CVE-2019-9512 e CVE-2019-9514 .	6 settembre 2019
Annuncio della definizione di Kubernetes1.11 come obsoleto in Amazon EKS	Amazon EKS ha interrotto il supporto per Kubernetes versione 1.11 il 4 novembre 2019.	4 settembre 2019
Kubernetes versione 1.14	Sono stati aggiunti il supporto per Kubernetes versione 1.14 per nuovi cluster e alcuni aggiornamenti della versione.	3 settembre 2019
Ruoli IAM per gli account di servizio	Grazie ai ruoli IAM per gli account di servizio sui cluster Amazon EKS è possibile associare un ruolo IAM a un account di servizio Kubernetes. Grazie a questa funzionalità, non è più necessario fornire autorizzazioni estese al ruolo IAM del nodo. In questo modo, Pods su quel nodo puoi chiamare le AWS API.	3 settembre 2019
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nella Regione AWS Medio Oriente (Bahrein) (me-south-1).	29 agosto 2019
Aggiornamento della versione della piattaforma Amazon EKS	Nuove versioni della piattaforma per gestire CVE-2019-9512 e CVE-2019-9514 .	28 agosto 2019

Aggiornamento della versione della piattaforma Amazon EKS	Nuove versioni della piattaforma per gestire CVE-2019-11247 e CVE-2019-11249 .	5 agosto 2019
Espansione delle Regioni di Amazon EKS	Amazon EKS è ora disponibile nella Regione AWS Asia Pacifico (Hong Kong) (ap-east-1).	31 luglio 2019
Kubernetes1.10 non è più supportato su Amazon EKS	Kubernetes versione 1.10 non è più supportato in Amazon EKS. Eseguire l'aggiornamento di ogni cluster 1.10 alla versione 1.11 o successiva per evitare interruzioni del servizio.	30 luglio 2019
Aggiunto argomento sul controller di ingresso ALB	L' AWS ALB Ingress Controller for Kubernetes è un controller che consente la creazione di un ALB quando vengono create risorse in ingresso.	11 luglio 2019
Nuova AMI ottimizzata per Amazon EKS	Rimozione del file binario kubectl superfluo dalle AMI.	3 luglio 2019
Kubernetes versione 1.13	Sono stati aggiunti il supporto per Kubernetes versione 1.13 per nuovi cluster e alcuni aggiornamenti della versione.	18 giugno 2019
Nuova AMI ottimizzata per Amazon EKS con patch per AWS-2019-005	Amazon EKS ha aggiornato l'AMI ottimizzata per Amazon EKS per gestire le vulnerabilità descritte in AWS-2019-005 .	17 giugno 2019

Annuncio della definizione di Kubernetes`1.10` come obsoleto in Amazon EKS	Amazon EKS ha smesso di supportare Kubernetes versione `1.10` il 22 luglio 2019.	21 maggio 2019
Aggiornamento della versione della piattaforma Amazon EKS	Nuova versione della piattaforma per cluster Kubernetes `1.11` e `1.10` per supportare nomi DNS personalizzati nel certificato `kubelet` e migliorare le prestazioni di `etcd`.	21 maggio 2019
Comando AWS CLI`get-token`	Il comando `aws eks get-token` stato aggiunto alla AWS CLI, Non è più necessario installare AWS IAM Authenticator per creare token di sicurezza client Kubernetes per la comunicazione tra server API del cluster. Aggiorna AWS CLI l'installazione alla versione più recente per utilizzare questa nuova funzionalità. Per ulteriori informazioni, consulta Installazione dell' AWS Command Line Interface nella Guida per l'utente dell'AWS Command Line Interface .	10 maggio 2019

Nozioni di base su `eksctl`	Questa guida alle operazioni di base ti consente di creare tutte le risorse necessarie per cominciare a utilizzare `eksctl` in Amazon EKS. Questa è una semplice utility a riga di comando per la creazione e la gestione di cluster Kubernetes su Amazon EKS.	10 maggio 2019
Aggiornamento della versione della piattaforma Amazon EKS	Nuova versione della piattaforma per cluster Kubernetes `1.12` per supportare nomi DNS personalizzati nel certificato `kubelet` e migliorare le prestazioni di `etcd`. La versione corregge un bug che costringeva i daemon `kubelet` dei nodi a richiedere un nuovo certificato a intervalli di pochi secondi.	8 maggio 2019
Tutorial su Prometheus	È stato aggiunto un argomento per l'implementazione di Prometheus nel tuo cluster Amazon EKS.	5 aprile 2019

Registrazione del piano di controllo Amazon EKS	Con questo aggiornamento è possibile ottenere registri di verifica e di diagnostica direttamente dal piano di controllo di Amazon EKS. È possibile utilizzare questi CloudWatch registri nel proprio account come riferimento per la protezione e l'esecuzione dei cluster.	4 aprile 2019
Kubernetes versione `1.12`	Sono stati aggiunti il supporto per Kubernetes versione `1.12` per nuovi cluster e alcuni aggiornamenti della versione.	28 marzo 2019
Aggiunta guida alle operazioni di base di App Mesh	È stata aggiunta la documentazione per le nozioni di base su App Mesh e Kubernetes.	27 marzo 2019
Accesso privato all'endpoint del server API di Amazon EKS	Aggiunta la documentazione per la disabilitazione dell'accesso pubblico per l'endpoint del server API Kubernetes del cluster Amazon EKS.	19 marzo 2019
È stato aggiunto l'argomento relativo all'installazione del server dei parametri Kubernetes	Il server dei parametri Kubernetes è un aggregatore dei dati di utilizzo delle risorse nel cluster.	18 marzo 2019
Aggiunto elenco di progetti open source correlati	Questi progetti open source estendono la funzionalità dei Kubernetes cluster in esecuzione AWS, inclusi i cluster gestiti da Amazon EKS.	15 marzo 2019

Aggiunto argomento per l'installazione di Helm in locale	Il programma di gestione del pacchetto `helm` per Kubernetes consente di installare e gestire le applicazioni su cluster Kubernetes. Questo argomento illustra come installare ed eseguire i binari `helm` e `tiller` localmente per consentire l'installazione e la gestione di grafici utilizzando la CLI helm nel sistema locale.	11 marzo 2019
Aggiornamento della versione della piattaforma Amazon EKS	Nuova versione della piattaforma per l'aggiornamento dei cluster Amazon EKS Kubernetes `1.11` al livello di patch `1.11.8` per gestire `CVE-2019-1002100` .	8 marzo 2019
Limite cluster aumentato	Amazon EKS ha innalzato il numero di cluster che è possibile creare in una Regione AWS da 3 a 50.	13 febbraio 2019
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile in Europa (Londra) (`eu-west-2`), Europa (Parigi) (`eu-west-3`) e Asia Pacifico (Mumbai) (`ap-south-1`) Regioni AWS.	13 febbraio 2019
Nuova AMI ottimizzata per Amazon EKS con patch per `ALAS-2019-1156`	Amazon EKS ha aggiornato l'AMI ottimizzata per Amazon EKS per affrontare la vulnerabilità descritta in `ALAS-2019-1156` .	11 febbraio 2019

Nuova AMI ottimizzata per Amazon EKS con patch per `ALAS2-2019-1141`	Amazon EKS ha aggiornato l'AMI ottimizzata per Amazon EKS per affrontare le problematiche di tipo CVE (Common Vulnerabilities & Exposures) riportate in `ALAS2-2019-1141`.	9 gennaio 2019
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nella regione Asia-Pacifico (Seoul) (`ap-northeast-2`). Regione AWS	9 gennaio 2019
Espansione delle Regioni di Amazon EKS	Amazon EKS è ora disponibile nei seguenti paesi aggiuntivi Regioni AWS: Europa (Francoforte) (`eu-central-1`), Asia Pacifico (Tokyo) (`ap-northeast-1`), Asia Pacifico (Singapore) (`ap-southeast-1`) () e Asia Pacifico (Sydney) (`ap-southeast-2`).	19 dicembre 2018
Aggiornamenti del cluster Amazon EKS	È stata aggiunta la documentazione relativa agli aggiornamenti della versione del cluster Kubernetes e per la sostituzione dei nodi per Amazon EKS.	12 dicembre 2018
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nella Regione AWS Europa (Stoccolma) (`eu-north-1`).	11 dicembre 2018

Aggiornamento della versione della piattaforma Amazon EKS	Nuova versione della piattaforma per l'aggiornamento di Kubernetes a livello di patch `1.10.11` per risolvere `CVE-2018-1002105` .	4 dicembre 2018
È stato aggiunto il supporto alla versione `1.0.0` per il controller in ingresso ALB	Il controller di ingresso ALB rilascia una versione `1.0.0` con supporto formale da. AWS	20 novembre 2018
Aggiunto supporto per la configurazione di rete CNI	Ora Amazon VPC CNI plugin for Kubernetes versione `1.2.1` supporta la configurazione di rete personalizzata per interfacce di rete di Pod secondari.	16 ottobre 2018
È stato aggiunto il supporto per `MutatingAdmissionWebhook` e `ValidatingAdmissionWebhook`	La versione della piattaforma Amazon EKS `1.10-eks.2` ora supporta i controller `MutatingAdmissionWebhook` e di ammissione `ValidatingAdmissionWebhook` .	10 ottobre 2018
Sono state aggiunte le informazioni dell'AMI del partner	Canonical ha collaborato con Amazon EKS per creare AMI del nodo utilizzabili nei cluster.	3 ottobre 2018
Aggiunte le istruzioni per il comando AWS CLI`update-kubeconfig`	Amazon EKS lo ha aggiunto `update-kubeconfig` per AWS CLI semplificare il processo di creazione di un `kubeconfig` file per l'accesso al cluster.	21 settembre 2018

Nuove AMI ottimizzate per Amazon EKS	Amazon EKS ha aggiornato le AMI ottimizzate per Amazon EKS (con e senza il supporto di GPU) per fornire diverse correzioni di sicurezza e ottimizzazioni delle AMI.	13 settembre 2018
Regione AWS Espansione di Amazon EKS	Amazon EKS è ora disponibile nella Regione UE (Irlanda) (`eu-west-1`).	5 settembre 2018
Aggiornamento della versione della piattaforma Amazon EKS	Nuova versione della piattaforma con supporto per il livello di aggregazione Kubernetes e per Horizontal Pod Autoscaler (HPA).	31 agosto 2018
Nuove AMI ottimizzate per Amazon EKS e supporto di GPU	Amazon EKS ha aggiornato l'AMI ottimizzata per Amazon EKS per utilizzare un nuovo modello di AWS CloudFormation nodo e uno script di bootstrap. Inoltre, è disponibile una nuova AMI ottimizzata per Amazon EKS con supporto di GPU.	22 agosto 2018
Nuova AMI ottimizzata per Amazon EKS con patch per `ALAS2-2018-1058`	Amazon EKS ha aggiornato l'AMI ottimizzata per Amazon EKS per affrontare le problematiche di tipo CVE (Common Vulnerabilities & Exposures) riportate in `ALAS2-2018-1058`.	14 agosto 2018

Script della build AMI ottimizzata per Amazon EKS	Amazon EKS ha reso open-source gli script di build che vengono utilizzati per creare l'AMI ottimizzata per Amazon EKS. Questi script di build sono ora disponibili su GitHub.	10 luglio 2018
Versione iniziale di Amazon EKS	Documentazione iniziale per l'avvio del servizio	5 giugno 2018

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.